KDJ11-B KDJ11-B CLUSTER DIAG
COKDAAQ

AH-T854A-MC
FICHE 1 OF 3

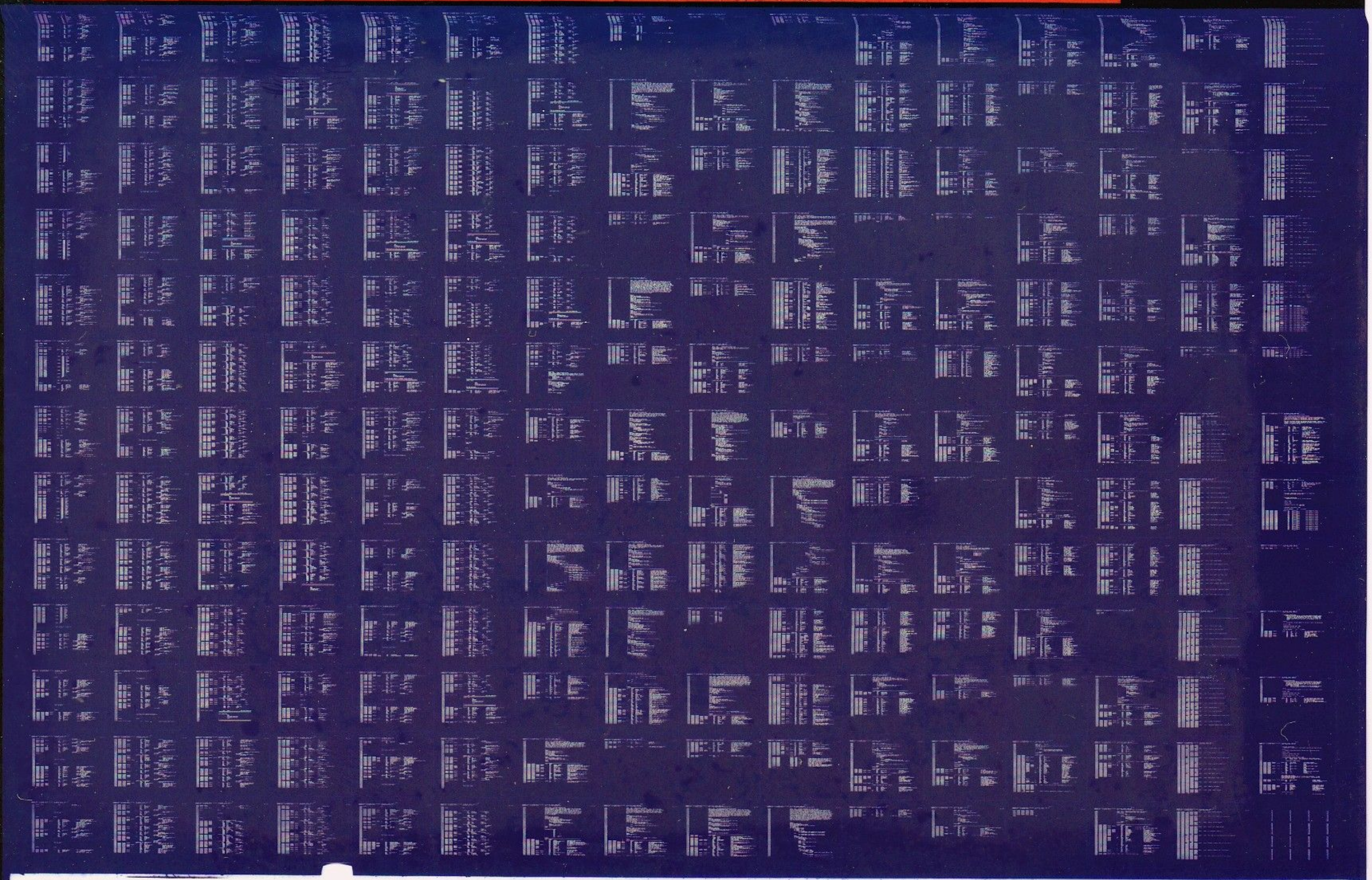
APR 1984
digital
Made In USA

KDJ11-B

KDJ11-B CLUSTER DIAG
COKDAAO

AH-T854A-MC
FICHE 2 OF 3

APR 1984
digital
Made In USA

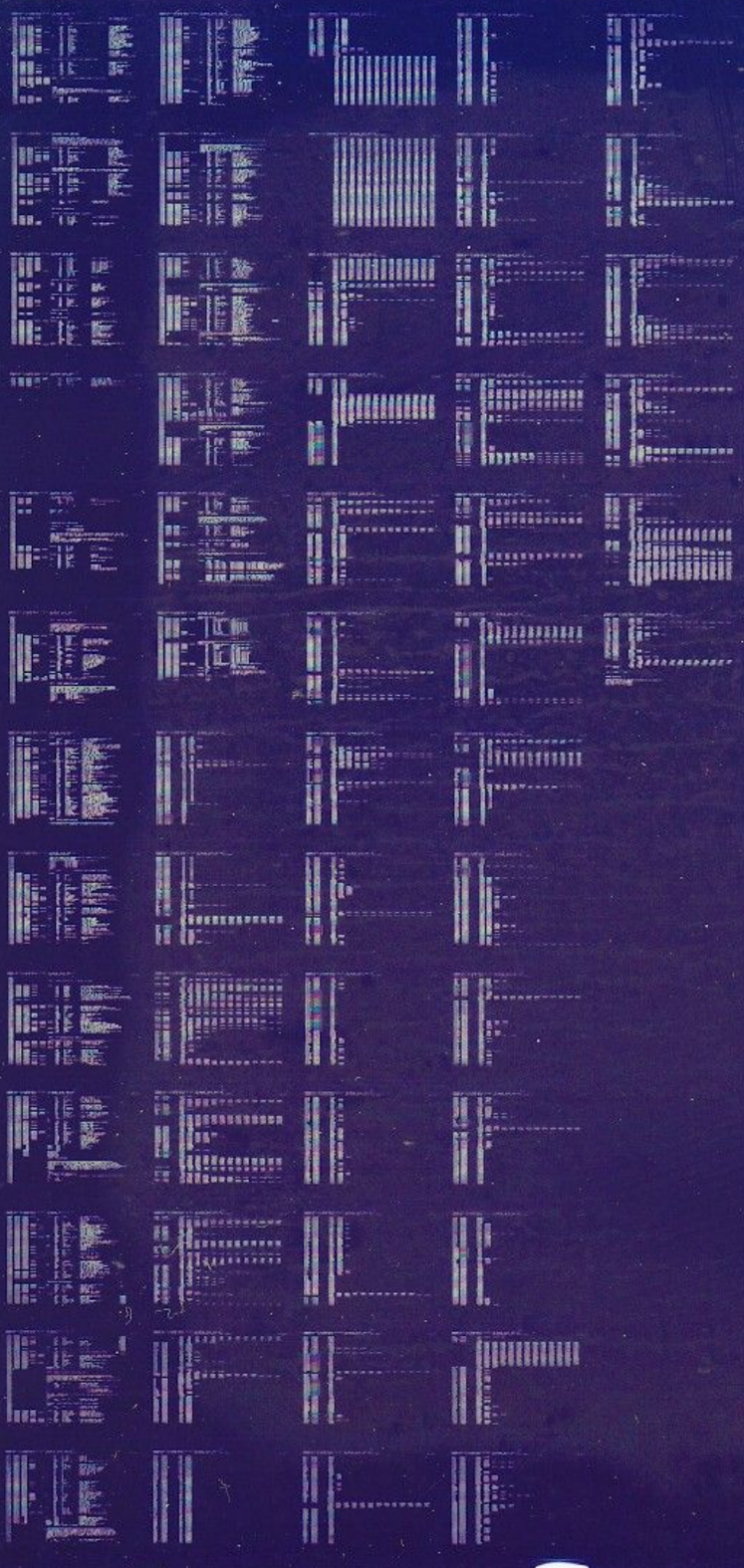


KDJ11-B

KDJ11-B CLUSTER DIAG
COKDAAO

AH-T854A-MC
FICHE 3 OF 3

APR 1984
digital
Made In USA



```
                                                        .REM &

                                IDENTIFICATION
                                --------------

                        PRODUCT CODE:  AC-T853A-MC

                        PRODUCT NAME:  COKDAAO KDJ11-B CLUSTER DIAG

                        PRODUCT DATE:  DECEMBER, 1983

                        MAINTAINER:    DIAGNOSTIC ENGINEERING
```

TABLE OF CONTENTS

```
                                1.   PROGRAM ABSTRACT


THIS PROGRAM TESTS OUT KDJ11-B CPU BOARD, INCLUDING THE J11 CHIP SET,
ON-BOARD CACHE, ON-BOARD ROM'S, SERIAL LINE UNIT, AND LINE TIME
CLOCKS.
THE KDJ11-B IS A PDP-11 CPU THAT INCORPORATES THE J11 CHIP SET AS THE
HEART OF THE PROCESSOR. IT IS A QUAD HEIGHT Q22 BUS MODULE. IT HAS
ON-BOARD CACHE, SOME OF THE FUNCTIONALITY OF THE CACHE IS HIDDEN INSIDE
THE J11 AND THE REST OF THE FUNCTIONS IMPLEMENTED IN TWO ON-BOARD GATE
ARRAYS. THE STORAGE CAPACITY OF THE CACHE IS 4K BYTES OF RAM, CALLED
DATA RAM'S. THE CACHE IS IMPLEMENTED AS A DIRECT MAPPED CACHE WITH
ADDRESS BITS 21 THROUGH 13 STORED IN A DIFFERENT SET OF RAM'S CALLED
TAG STORE.
THE KDJ11-B ALSO HAS TWO ON-BOARD ROM'S. ONE OF THEM, THE 16-BIT
ADDRESSABLE ROM, CONTAINS THE SELF-TEST AND THE BOOT CODES. THE OTHER
ROM, THE 8-BIT ADDRESSABLE ONE, CONTAINS THE BASE AREA WITH HARDWARE
SELECTION PARAMETERS, OPTIONAL BOOTSTRAPS, OPTIONAL UFD (USER FRIENDLY
DIAGNOSTIC) SYSTEM DESCTIPTION AREA, AND OPTIONAL FOREIGN LANGUAGE
TEXT.
THE SERIAL LINE UNIT IS IMPLEMENTED THRU A DLART CHIP WHICH PROVIDES
THE STANDARD CONSOLE INTERFACE TO THE CPU. IT HAS INTERNAL LOOP BACK
MODE AND PROVIDES WITH THREE CLOCK LINES: 800HZ, 60HZ, AND 50HZ.
THE LINE TIME CLOCK FUNCTIONS ARE IMPLEMENTED USING THOSE THREE LINES
AND THE BEVENT LINE. THE LINE CLOCK STATUS REGISTER IS IMPLEMENTED IN
ONE OF THE GATE ARRAYS.


                                2.  SYSTEM REQUIREMENTS

                                HARDWARE REQUIREMENTS

                                TO RUN SUCCESSFULLY THE DIAGNOSTIC NEEDS:

                                        1. KDJ11-B CPU MODULE
                                        2. CONSOLE TERMINAL
                                        3. AT LEAST 28K OF MEMORY

                                IN DVT,  AND STAGE ONE MANUFACTURING (MODULE ASSEMBLY) THE Q22 BUS
                                EXERCISER IS NEEDED TO CHECK Q22 BUS LOGIC.


                                3.  LOADING AND STARTING PROCEDURES

                                TO START-UP THIS PROGRAM:

                                        1. BOOT XXDP+

                                        2. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC
                                           FILE FOR THIS PROGRAM.

                                THE STARTING ADDRESS OF THE PROGRAM IS 200.
                                NOTE: IF TRYING TO RESTART THE PROGRAM IN AN ARBITRARY PLACE AFTER
                                HALT ON BREAK THE FOLLOWING REGISTERS SHOULD BE SET UP:
                                17777572=0      TO DISABLE MEMMORY MANAGEMENT
```

17777520=1000 TO CLEAR DIAGNOSTIC MODE (BIT 8), BUT STILL SAVE
               HALT ON BREAK
17777746=400   TO FLUSH THE CACHE

4. SPECIAL ENVIRONMENTS

THE PROGRAM IS APT COMPATIBLE. IT CAN ALSO BE RUN UNDER THE UFD MONITOR.
IN THOSE CASES NONE OF THE STANDARD ERROR PRINTOUTS OCCUR. REFER
TO CORRESPONDING DOCUMENTS ON RUNNING PROCEDURES IN APT AND UNDER
UFD MONITOR.

5. PROGRAM OPTIONS

THE Q22 BUS EXERCISER IS UTILIZED IF IT IS PRESENT IN THE SYSTEM AND
THE DIAGNOSTIC IS NOT RUNNING IN UFD MODE.
STANDARD CAPABILITIES OF LOOPING ON TEST AND ON ERROR ARE PROVIDED.
IN ORDER TO RUN THE EXTENSIVE CACHE DATA RAM TEST BIT 7 HAS TO BE SET
IN THE SOFTWARE SWITCH REGISTER. TO RUN THE EXTENSIVE CACHE TAG RAM TEST
BIT 6 HAS TO BE SET IN THE SOFTWARE SWITCH REGISTER.

SWITCH REGISTER SELECTION:

BIT NUMBER	USE
15	HALT ON ERROR
14	LOOP ON PRESENT TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<5-0>
7	DO EXTENSIVE DATA RAM TEST
6	DO EXTENSIVE TAG RAM TEST

6. EXECUTION TIMES

WITHOUT EXTENSIVE RAM TESTS, THE DIAGNOSTIC RUNS IN UNDER 15 SECONDS.
WITH THEM, IT TAKES ABOUT 2 MINUTES.

7. ERROR INFORMATION

IN THE CASE OF ERRORS, A FAILING PC AND TEST NUMBERS ARE GIVEN.
WHERE IT IS POSSIBLE, EXPECTED AND RECEIVED DATA ARE GIVEN.
FOR AN EXAMPLE, SEE SECTION 8.

8. EXAMPLES

AFTER BOOTING XXDP+ AND STARTING THE PROGRAM, THE FOLLOWING
WILL APPEAR ON THE TERMINAL:

```
KDJ11-B CPU DIAGNOSTIC

SWR = XXXXXX    NEW =
```

WHERE XXXXXX CORRESPOND TO PRESENT SOFTWARE SWITCH REGISTER

```
186                                     SETTING.
187                                     AFTER "NEW" AN OPERATOR CAN DO ONE OF THE FOLLOWING:
188                                     1) TYPE IN A NEW SOFTWARE SWITCH REGISTER SETTING FOLLOWED
189                                     BY CARRIGE RETURN OR
190                                     2) JUST TYPE IN CARRIAGE RETURN IN WHICH CASE THE SOFTWARE
191                                     REGISTER WILL REMAIN UNCHANGED.
192
193                                     EXAMPLE OF ERROR PRINTOUT:
194                                             ERROR IN TAG STORE
195                                              TEST   ERROR   ADDRESS ADDRESS
196                                               #      PC     <21-16> <15-0>
197                                              27    105620  66600   000000
198                                     NOTE: THIS MAY NOT CORRESPOND TO THE ACTUAL PROGRAM COUNTER.
199
200                                     9. PROGRAM DESCRIPTION
201
202                                     9.1  J11 CODE
203                                     THIS PORTION OF THE CODE TESTS OUT THE J11 CHIP SET. IT IS BROKEN
204                                     INTO 3 PIECES: CPU TESTS, WHICH VERIFY DIFFERENT INSTRUCTIONS IN
205                                     DIFFERENT MODES AND DIFFERENT TRAP CONDITIONS; MMU TESTS, WHICH
206                                     VERIFY DIFFERENT FUNCTIONS OF MMU; AND FFP TESTS, WHICH DO
207                                     DIFFERENT FLOATING POINT INSTRUCTIONS.
208                                     THIS PORTION OF THE CODE HAVE BEEN WRITTEN IN CLOSE RELATIONSHIP
209                                     WITH THE J11 MICROCODE. THEREFORE, EVEN THOUGH NOT ALL POSSIBLE
210                                     INSTRUCTIONS IN ALL POSSIBLE ADDRESSING MODE HAVE BEEN TESTED, AN
211                                     ATTEMPT HAS BEEN MADE TO EXERCISE ALL OF THE MICROCODE.
212
213                                     9.2  CACHE CODE
214                                     THIS PORTION OF THE DIAGNOSTIC VERIFIES ALL CACHE FUNCTIONS.
215                                     DATA AND TAG RAM'S TESTS ARE ALSO INCLUDED.
216
217                             ********************************************************************************
218                                     NOTE: IN ORDER TO RUN EXTENSIVE CACHE RAM'S TESTS THE CORRESPONDING
219                                     BITS IN THE SOFTWARE SWITCH REGISTER SHOULD BE SET. SEE SECTION 5.
220                             ********************************************************************************
221
222                                     9.3  ON-BOARD ROM'S CODE
223                                     THESE TESTS VERIFY THE CHECKSUMS OF THE 16-BIT ROM AND THE BASE
224                                     AREA OF THE 8-BIT ROM.
225
226                                     9.4  LINE TIME CLOCKS CODE
227                                     THIS PART OF THE PROGRAM VERIFIES THE FUNCTIONS OF ALL 4 CLOCK
228                                     LINES: 3 OF THEM FROM THE SERIAL LINE CHIP (50HZ, 60HZ, 800HZ)
229                                     AND BEVENT LINE.
230                                     NOTE: IN UFD MODE ONLY FUNCTIONS CORRESPONDING TO BOOT ROM
231                                     SELECTION ARE CHECKED.
232
233                                     9.5  SERIAL LINE UNIT CODE
234                                     THESE TESTS VERIFY THE FUNCTIONALITY OF THE SLU CHIP UTILIZING THE
235                                     MAINTENANCE MODE OF THE CHIP.
236
```

```
237                                     9.6  Q22BE CODE
238                                     THESE TESTS VERIFY INTERRUPT ARBITRATION OF THE KDJ11 B, DMA
239                                     PROTOCOL, AND CACHE FUNCTIONS RELATED TO THE DMA ACTIVITIES,
240                                     INCLUDING PMG COUNTER.
241
242                                                                                             ε
243
```

```
                        167400                          $SWR=167400
                        000300                          $SWRMK=300
                                                .TITLE  COKDAAO KDJ11-B CLUSTER
                                                ;*COPYRIGHT (C) DEC 83
                                                ;*DIGITAL EQUIPMENT CORP.
                                                ;*MAYNARD, MASS. 01754
                                                ;*
                                                ;*PROGRAM BY DIAG. ENG.
                                                ;*
                                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                                ;*PACKAGE (MAINDEC-11-DZQAC-C8), OCT, 1982.
                                                ;*
                        000001                  $TN=1
                                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                                ;*
                                                ;*      SWITCH                  USE
                                                ;*      ------          --------------------
                                                ;*        15            HALT ON ERROR
                                                ;*        14            LOOP ON TEST
                                                ;*        13            INHIBIT ERROR TYPEOUTS
                                                ;*        11            INHIBIT ITERATIONS
                                                ;*        10            BELL ON ERROR
                                                ;*         9            LOOP ON ERROR
                                                ;*         8            LOOP ON TEST IN SWR<5:0>
                                                ;*         7            DO EXTENSIVE DATA RAM TEST
                                                ;*         6            DO EXTENSIVE TAG RAM TEST
                                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                                                ;*KT11 VECTOR ADDRESS

                        000250                  MMVEC=  250

                                                ;*KT11 STATUS REGISTER ADDRESSES

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

                                                ;*USER "I" PAGE DESCRIPTOR REGISTERS

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

                                                ;*USER "D" PAGE DESCRIPTOR REGISTORS

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

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

                                        ;*USER "I" PAGE ADDRESS REGISTERS

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

                                        ;*USER "D" PAGE ADDRESS REGISTERS

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

                                        ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS

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

                                        ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS

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

                                        ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS

                172240                  SIPAR0= 172240
                172242                  SIPAR1= 172242
                172244                  SIPAR2= 172244
                172246                  SIPAR3= 172246
                172250                  SIPAR4= 172250
```

```
                        172252          SIPAR5= 172252
                        172254          SIPAR6= 172254
                        172256          SIPAR7= 172256

                                        ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

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

                                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                                        ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

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

                                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

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

                                        ;*KERNEL "D" PAGE ADDRESS REGISTERS

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

```
                        KDPAR6= 172374
                        KDPAR7= 172376

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
104000                  ERROR=  EMT             ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE=  IOT             ;;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
177776                  PSW=    PS
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

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

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
```

L1


```
   468          000020                  SW04=   20
   469          000010                  SW03=   10
   470          000004                  SW02=   4
   471          000002                  SW01=   2
   472          000001                  SW00=   1
   473          001000                  SW9=    SW09
   474          000400                  SW8=    SW08
   475          000200                  SW7=    SW07
   476          000100                  SW6=    SW06
   477          000040                  SW5=    SW05
   478          000020                  SW4=    SW04
   479          000010                  SW3=    SW03
   480          000004                  SW2=    SW02
   481          000002                  SW1=    SW01
   482          000001                  SW0=    SW00
   483
   484                                  ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
   485          100000                  BIT15=  100000
   486          040000                  BIT14=  40000
   487          020000                  BIT13=  20000
   488          010000                  BIT12=  10000
   489          004000                  BIT11=  4000
   490          002000                  BIT10=  2000
   491          001000                  BIT09=  1000
   492          000400                  BIT08=  400
   493          000200                  BIT07=  200
   494          000100                  BIT06=  100
   495          000040                  BIT05=  40
   496          000020                  BIT04=  20
   497          000010                  BIT03=  10
   498          000004                  BIT02=  4
   499          000002                  BIT01=  2
   500          000001                  BIT00=  1
   501          001000                  BIT9=   BIT09
   502          000400                  BIT8=   BIT08
   503          000200                  BIT7=   BIT07
   504          000100                  BIT6=   BIT06
   505          000040                  BIT5=   BIT05
   506          000020                  BIT4=   BIT04
   507          000010                  BIT3=   BIT03
   508          000004                  BIT2=   BIT02
   509          000002                  BIT1=   BIT01
   510          000001                  BIT0=   BIT00
   511
   512                                  ;*BASIC "CPU" TRAP VECTOR ADDRESSES
   513          000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
   514          000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
   515          000014                  TBITVEC=14              ;;"T" BIT
   516          000014                  TRTVEC= 14              ;;TRACE TRAP
   517          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
   518          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
   519          000024                  PWRVEC= 24              ;;POWER FAIL
   520          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
   521          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   522          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   523          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
```

```
                000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                000001                          UFDSET=         1               ;FLAG FOR UFD
                                        .SBTTL  TRAP CATCHER

                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                          .=174
000174  000000                         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                000200                  .=200
000200  005037  001160                          CLR     $TMP0
000204  000137  003764                          JMP     @#START
                000220                  .=220
000220  012737  000777  001160                  MOV     #777,$TMP0
000226  000137  003764                          JMP     @#START

                                        .SBTTL  ACT11 HOOKS

                                        ;;**************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000232                          $SVPC=.                 ;SAVE PC
                000046                          .=46
000046  133020                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                          .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                000232                          .=$SVPC                 ;; RESTORE PC
                                        .SBTTL  APT PARAMETER BLOCK

                                        ;;**************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**************************************************************
                000232                  .$X=.   ;;SAVE CURRENT LOCATION
                000024                  .=24    ;;SET POW R FAIL TO POINT TO START OF PROGRAM
000024  000200                         200     ;;FOR AP  START UP
                000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  000232                          $APTHDR ;;POINT TO APT HEADER BLOCK
                000232                  .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.

000232                                  $APTHD:
000232  000000                          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000234  001200                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
000236  000000                          $TSTM:  .WORD           ;;RUN TIM OF LONGEST TEST
000240  000000                          $PASTM: .WORD           ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000242  000000                          $UNITM: .WORD           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
000244  000052                                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

            001100                      .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102     000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134     000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $TMP0:  .WORD   0               ;;USER DEFINED
001162  000000                  $TMP1:  .WORD   0               ;;USER DEFINED
001164  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001166  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001170  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001174     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
001175     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001176  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001200                          $MAIL:                  ;;APT MAILBOX
001200  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001202  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
001204  000000                  $TESTN: .WORD   ATESTN   ;;TEST NUMBER
001206  000000                  $PASS:  .WORD   APASS    ;;PASS COUNT
001210  000000                  $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
```

```
   630  001212  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
   631  001214  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
   632  001216  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
   633  001220                          $ETABLE:                ;;APT ENVIRONMENT TABLE
   634  001220     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
   635  001221     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
   636  001222  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
   637  001224  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
   638  001226  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
   639                                  ;*                      BITS 15-11=CPU TYPE
   640                                  ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
   641                                  ;*                              11/70=06,PDQ=07,Q=10
   642                                  ;*                      BIT 10=REAL TIME CLOCK
   643                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
   644                                  ;*                      BIT  8=MEMORY MANAGEMENT
   645  001230     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
   646  001231     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
   647                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
   648                                  ;*                              900 NSEC CORE=001
   649                                  ;*                              300 NSEC BIPOLAR=002
   650                                  ;*                              500 NSEC MOS=003
   651  001232  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
   652                                  ;*                        MEM.LAST ADDR.-3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
   653  001234     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
   654  001235     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
   655  001236  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
   656  001240     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
   657  001241     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
   658  001242  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
   659  001244     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
   660  001245     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
   661  001246  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
   662  001250  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
   663  001252  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
   664  001254  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
   665  001256  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
   666  001260  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
   667  001262  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
   668  001264  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
   669  001266  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
   670  001270  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
   671  001272  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
   672  001274  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
   673  001276  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
   674  001300  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
   675  001302  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
   676  001304  000000                  $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
   677  001306  000000                  $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
   678  001310  000000                  $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
   679  001312  000000                  $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
   680  001314  000000                  $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
   681  001316  000000                  $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
   682  001320  000000                  $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
   683  001322  000000                  $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
   684
   685
```

```
   686  001324                          $ETEND:
   687
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001324                          $ERRTB:


                                .SBTTL  ERROR DEFINITIONS
                                ;ITEM 1
001324  122020                          EM1             ;CPU ERROR
001326  127564                          DH1             ;TEST #, ERROR PC
001330  130634                          DT1             ;$TMP1,$ERRPC
001332  000000                          0
                                ;ITEM 2
001334  122054                          EM2             ;MMU ERROR
001336  127564                          DH1             ;TEST #, ERROR PC
001340  130634                          DT1             ;$TMP1,$ERRPC
001342  000000                          0

                                ;ITEM 3
001344  122066                          EM3             ;FPP ERROR
001346  127564                          DH1             ;TEST #, ERROR PC
001350  130634                          DT1             ;$TMP1,$ERRPC
001352  000000                          0

                                ;ITEM 4
001354  122100                          EM4             ;ERROR IN READ-WRITE BITS OF CCR
001356  127611                          DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
001360  130642                          DT4             ;$TMP1,$ERRPC,R1,CCR
001362  000000                          0

                                ;ITEM 5
001364  122140                          EM5             ;FORCE MISS WRITES TO CACHE
001366  127676                          DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
001370  130654                          DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
001372  000000                          0

                                ;ITEM 6
001374  122173                          EM6             ;FORCE MISS WRITE INVALIDATES CACHE
001376  127676                          DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
001400  130654                          DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
001402  000000                          0

                                ;ITEM 7
001404  122236                          EM7             ;UNEXPECTED PARITY INTERRUPT
001406  127775                          DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
001410  130670                          DT7             ;$TMP1,$ERRPC,$BDADR,MSER
001412  000000                          0

                                ;ITEM 10
001414  122272                          EM10            ;TAG PARITY ERROR
001416  127775                          DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
```

E2


```
744  001420  130670                          DT7             ;$TMP1,$ERRPC,$BDADR,MSER
745  001422  000000                          0
746                                  ;ITEM 11
747  001424  122313                          EM11            ;DATA PARITY ERROR
748  001426  127775                          DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
749  001430  130670                          DT7             ;$TMP1,$ERRPC,$BDADR,MSER
750  001432  000000                          0
751                                  ;ITEM 12
752  001434  122335                          EM12            ;LOW BYTE PARITY ERROR
753  001436  127775                          DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
754  001440  130670                          DT7             ;$TMP1,$ERRPC,$BDADR,MSER
755  001442  000000                          0
756                                  ;ITEM 13
757  001444  122363                          EM13            ;HIGH BYTE PARITY ERROR
758  001446  127775                          DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
759  001450  130670                          DT7             ;$TMP1,$ERRPC,$BDADR,MSER
760  001452  000000                          0
761                                  ;ITEM 14
762  001454  122412                          EM14            ;ERROR DATA PATH
763  001456  127611                          DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
764  001460  130702                          DT14            ;$TMP1,$ERRPC,$GDDAT,TSTLOC
765  001462  000000                          0
766                                  ;ITEM 15
767  001464  122435                          EM15            ;FORCE MISS READS FROM CACHE
768  001466  127676                          DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
769  001470  130654                          DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
770  001472  000000                          0
771                                  ;ITEM 16
772  001474  122471                          EM16            ;FORCE MISS READS FROM CACHE AND MISS
773  001476  127676                          DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
774  001500  130654                          DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
775  001502  000000                          0
776                                  ;ITEM 17
777  001504  122536                          EM17            ;ERROR IN RECORDING HITS IN HIT/MISS
778  001506  127611                          DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
779  001510  130714                          DT17            ;$TMP1,$ERRPC,$GDDAT,RECDAT
780  001512  000000                          0
781                                  ;ITEM 20
782  001514  122602                          EM20            ;WRITE BYTE ALLOCATES CACHE
783  001516  127564                          DH1             ;TEST #, PC
784  001520  130634                          DT1             ;$TMP1,$ERRPC
785  001522  000000                          0
786                                  ;ITEM 21
787  001524  122635                          EM21            ;WRITE BYTE HIT DOES NOT RECORD HIT
788  001526  127564                          DH1             ;TEST #, PC
789  001530  130634                          DT1             ;$TMP1,$ERRPC
790  001532  000000                          0
791                                  ;ITEM 22
792  001534  122700                          EM22            ;BYTES REVERSED ON WRITE CYCLES
793  001536  127564                          DH1             ;TEST #, PC
794  001540  130634                          DT1             ;$TMP1,$ERRPC
795  001542  000000                          0
796                                  ;ITEM 23
797  001544  122737                          EM23            ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
798  001546  127676                          DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
799  001550  130654                          DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
```

F2


```
  800  001552  000000                          0
  801                                ;ITEM 24
  802  001554  123013                          EM24            ;HITS RECORDED AFTER FLUSHING CACHE
  803  001556  130050                          DH24            ;TEST #, PC, NUMBER OF HITS
  804  001560  130726                          DT24            ;$TMP1,$ERRPC,R3
  805  001562  000000                          0
  806                                ;ITEM 25
  807  001564  123056                          EM25            ;BYPASS DOESN'T INVALIDATE CACHE
  808  001566  127564                          DH1             ;TEST #, PC
  809  001570  130634                          DT1             ;$TMP1,$ERRPC
  810  001572  000000                          0
  811                                ;ITEM 26
  812  001574  123116                          EM26            ;MSER DOES NOT CLEAR ON WRITE REFERENCE
  813  001576  127564                          DH1             ;TEST #, PC
  814  001600  130634                          DT1             ;$TMP1,$ERRPC
  815  001602  000000                          0
  816                                ;ITEM 27
  817  001604  123165                          EM27            ;PARITY ERROR DON'T CAUSE A MISS
  818  001606  130114                          DH27            ;TEST #, PC, MSER, HIT/MISS
  819  001610  130736                          DT27            ;$TMP1,$ERRPC,MSER,R3,0
  820  001612  000000                          0
  821                                ;ITEM 30
  822  001614  123225                          EM30            ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
  823  001616  130114                          DH27            ;TEST #, PC, MSER, HIT/MISS
  824  001620  130736                          DT27            ;$TMP1,$ERRPC,MSER,R3,0
  825  001622  000000                          0
  826                                ;ITEM 31
  827  001624  123277                          EM31            ;PARITY ERROR IGNORED
  828  001626  127564                          DH1             ;TEST #, PC
  829  001630  130634                          DT1             ;$TMP1,$ERRPC
  830  001632  000000                          0
  831                                ;ITEM 32
  832  001634  123324                          EM32            ;PARITY ERROR IGNORED ON LOW BYTE
  833  001636  127564                          DH1             ;TEST #, PC
  834  001640  130634                          DT1             ;$TMP1,$ERRPC
  835  001642  000000                          0
  836                                ;ITEM 33
  837  001644  123365                          EM33            ;PARITY ERROR IGNORED ON HIGH BYTE
  838  001646  127564                          DH1             ;TEST #, PC
  839  001650  130634                          DT1             ;$TMP1,$ERRPC
  840  001652  000000                          0
  841                                ;ITEM 34
  842  001654  123427                          EM34            ;PARITY ABORT LOGIC DOESN'T WORK
  843  001656  127564                          DH1             ;TEST #, PC
  844  001660  130634                          DT1             ;$TMP1,$ERRPC
  845  001662  000000                          0
  846                                ;ITEM 35
  847  001664  123467                          EM35            ;MSER NOT SET PROPERLY
  848  001666  127611                          DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
  849  001670  130750                          DT35            ;$TMP1,$ERRPC,$GDDAT,MSER,0
  850  001672  000000                          0
  851                                ;ITEM 36
  852  001674  123515                          EM36            ;PARITY INTERRUPT DOESN'T WORK
  853  001676  127564                          DH1             ;TEST #, PC
  854  001700  130634                          DT1             ;$TMP1,$ERRPC
  855  001702  000000                          0
```

```
856
857  001704  123561                    ;ITEM 37
858  001706  127564                            EM37            ;NXM AND PARITY ABORT DIN'T HAPPEN
859  001710  130634                            DH1             ;TEST #, PC
860  001712  000000                            DT1             ;$TMP1,$ERRPC
861                                             0
862  001714  123623                    ;ITEM 40
863  001716  127564                            EM40            ;PARITY ABORT NOT BLOCKED BY NXM TRAP
864  001720  130634                            DH1             ;TEST #, PC
865  001722  000000                            DT1             ;$TMP1,$ERRPC
866                                             0
867  001724  123670                    ;ITEM 41
868  001726  130157                            EM41            ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS
869  001730  130762                            DH41            ;TEST #, PC, INSTRUCTION OPCODE
870  001732  000000                            DT41            ;$TMP1,$ERRPC,$BDDAT
871                                             0
872  001734  123754                    ;ITEM 42
873  001736  127564                            EM42            ;ERROR IN PARITY LOGIC
874  001740  130634                            DH1             ;TEST #, PC
875  001742  000000                            DT1             ;$TMP1,$ERRPC
876                                             0
877  001744  124002                    ;ITEM 43
878  001746  130230                            EM43            ;ERROR IN CACHE DATA RAMS
879  001750  130772                            DH43            ;TEST #, PC, EXPECTED DATA, RECEIVED DATA, CACHE LOCATION
880  001752  000000                            DT43            ;$TMP1,$ERRPC,R1,RECDAT,$BDADR
881                                             0
882  001754  124033                    ;ITEM 44
883  001756  127564                            EM44            ;ERROR IN NXM IN STANDALONE MODE
884  001760  130634                            DH1             ;TEST #, PC
885  001762  000000                            DT1             ;$TMP1,$ERRPC
886                                             0
887  001764  124073                    ;ITEM 45
888  001766  127564                            EM45            ;HITS NOT RECORDED PROPERLY THRU HIT/MISS
889  001770  130634                            DH1             ;TEST #, PC
890  001772  000000                            DT1             ;$TMP1,$ERRPC
891                                             0
892  001774  124155                    ;ITEM 46
893  001776  130330                            EM46            ;HIT RECORDED FOR A LOCATION NOT IN CACHE
894  002000  131006                            DH47            ;TEST #, PC, LOCATION ACCESSED
895  002002  000000                            DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
896                                             0
897  002004  124245                    ;ITEM 47
898  002006  130330                            EM47            ;MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE
899  002010  131006                            DH47            ;TEST #, PC, LOCATION ACCESSED
900  002012  000000                            DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
901                                             0
902  002014  124332                    ;ITEM 50
903  002016  130330                            EM50            ;ERROR IN TAG STORE
904  002020  131020                            DH47            ;TEST #, PC, ADDRESS
905  002022  000000                            DT50            ;$TMP1,$ERRPC,R1,$BDADR
906                                             0
907  002024  124355                    ;ITEM 51
908  002026  127611                            EM51            ;ERROR PCR READ-WRITE BITS
909  002030  131032                            DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
910  002032  000000                            DT51            ;$TMP1,$ERRPC,$GDDAT,PCR
911                                             0
                                   ;ITEM 52
```

H2


```
  912  002034  124407                    EM52            ;ERROR IN BCSR READ-WRITE BITS
  913  002036  127611                    DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
  914  002040  131044                    DT52            ;$TMP1,$ERROC,$GDDAT,BCSR
  915  002042  000000                    0
  916                              ;ITEM 53
  917  002044  124445                    EM53            ;RESET DOESN'T CLEAR BCSR<4>
  918  002046  127564                    DH1             ;TEST #, PC
  919  002050  130634                    DT1             ;$TMP1,$ERRPC
  920  002052  000000                    0
  921                              ;ITEM 54
  922  002054  124501                    EM54            ;CHECKSUM ERROR IN 16-BIT ROM
  923  002056  127564                    DH1             ;TEST #, PC
  924  002060  130634                    DT1             ;$TMP1,$ERRPC
  925  002062  000000                    0
  926                              ;ITEM 55
  927  002064  124537                    EM55            ;CHCKSUM ERROR IN 8-BIT ROM
  928  002066  127564                    DH1             ;TEST #, PC
  929  002070  130634                    DT1             ;$TMP1,$ERRPC
  930  002072  000000                    0
  931                              ;ITEM 56
  932  002074  124573                    EM56            ;TIMEOUT READING LKS
  933  002076  127564                    DH1             ;TEST #, PC
  934  002100  130634                    DT1             ;$TMP1,$ERRPC
  935  002102  000000                    0
  936                              ;ITEM 57
  937  002104  124617                    EM57            ;LKS<07> DOES NOT BECOME 1
  938  002106  127564                    DH1             ;TEST #, PC
  939  002110  130634                    DT1             ;$TMP1,$ERRPC
  940  002112  000000                    0
  941                              ;ITEM 60
  942  002114  124651                    EM60            ;WRITE REFERENCE DOESN'T CLEAR LKS<07>
  943  002116  127564                    DH1             ;TEST #, PC
  944  002120  130634                    DT1             ;$TMP1,$ERRPC
  945  002122  000000                    0
  946                              ;ITEM 61
  947  002124  124717                    EM61            ;ILLEGAL LKS INTERRUPTS
  948  002126  127564                    DH1             ;TEST #, PC
  949  002130  130634                    DT1             ;$TMP1,$ERRPC
  950  002132  000000                    0
  951                              ;ITEM 62
  952  002134  124746                    EM62            ;PROCESSOR INTERRUPTS DON'T CLEAR LK <07>
  953  002136  127564                    DH1             ;TEST #, PC
  954  002140  130634                    DT1             ;$TMP1,$ERRPC
  955  002142  000000                    0
  956                              ;ITEM 63
  957  002144  125017                    EM63            ;LKS READY DOESN'T GO LOW
  958  002146  127564                    DH1             ;TEST #, PC
  959  002150  130634                    DT1             ;$TMP1,$ERRPC
  960  002152  000000                    0
  961                              ;ITEM 64
  962  002154  125050                    EM64            ;WRONG NUMBER OF LKS INTERRUPTS
  963  002156  127564                    DH1             ;TEST #, PC
  964  002160  130634                    DT1             ;$TMP1,$ERRPC
  965  002162  000000                    0
  966                              ;ITEM 65
  967  002164  125107                    EM65            ;LKS INTERRUPTS HAPPEN AT WRONG PRIORITY
```

```
  968  002166  130415                    DH65            ;TEST #, PC, PRIORITY
  969  002170  131070                    DT65            ;$TMP1,$ERRPC,$GDDAT
  970  002172  000000                    0
  971                              ;ITEM 66
  972  002174  125157                    EM66            ;BCSR<12> DOES NOT DISABLES LKS
  973  002176  127564                    DH1             ;TEST #, PC
  974  002200  130634                    DT1             ;$TMP1,$ERRPC
  975  002202  000000                    0
  976                              ;ITEM 67
  977  002204  125216                    EM67            ;BCSR<13> DOESN'T SET LKS<06>
  978  002206  127564                    DH1             ;TEST #, PC
  979  002210  130634                    DT1             ;$TMP1,$ERRPC
  980  002212  000000                    0
  981                              ;ITEM 70
  982  002214  125253                    EM70            ;RESET DOESN'T SET LKS<7>
  983  002216  127564                    DH1             ;TEST #, PC
  984  002220  130634                    DT1             ;$TMP1,$ERRPC
  985  002222  000000                    0
  986                              ;ITEM 71
  987  002224  125304                    EM71            ;RESET DOESN'T CLEAR LKS<06>
  988  002226  127564                    DH1             ;TEST #, PC
  989  002230  130634                    DT1             ;$TMP1,$ERRPC
  990  002232  000000                    0
  991                              ;ITEM 72
  992  002234  125340                    EM72            ;TIMEOUT READING SLU REGISTERS
  993  002236  130461                    DH72            ;TEST #, PC, ADDRESS FAILED
  994  002240  130762                    DT41            ;$TMP1,$ERRPC,$BDDAT
  995  002242  000000                    0
  996                              ;ITEM 73
  997  002244  125376                    EM73            ;XMIT READY DIDN'T GO LOW
  998  002246  127564                    DH1             ;TEST #, PC
  999  002250  130634                    DT1             ;$TMP1,$ERRPC
 1000  002252  000000                    0
 1001                              ;ITEM 74
 1002  002254  125422                    EM74            ;RCSR DOESN'T BECOME 1
 1003  002256  127564                    DH1             ;TEST #, PC
 1004  002260  130634                    DT1             ;$TMP1,$ERRPC
 1005  002262  000000                    0
 1006                              ;ITEM 75
 1007  002264  125453                    EM75            ;WRONG CHARACTER RECEIVED
 1008  002266  127611                    DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
 1009  002270  131100                    DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
 1010  002272  000000                    0
 1011                              ;ITEM 76
 1012  002274  125504                    EM76            ;RCSR<07> NOT CLEARED AFTER READING RBUF
 1013  002276  127564                    DH1             ;TEST #, PC
 1014  002300  130634                    DT1             ;$TMP1,$ERRPC
 1015  002302  000000                    0
 1016                              ;ITEM 77
 1017  002304  125554                    EM77            ;XCSR<07> NOT SET ON RESET
 1018  002306  127564                    DH1             ;TEST #, PC
 1019  002310  130634                    DT1             ;$TMP1,$ERRPC
 1020  002312  000000                    0
 1021                              ;ITEM 100
 1022  002314  125606                    EM100           ;RCSR<07> NOT CLEARED ON RESET
 1023  002316  127564                    DH1             ;TEST #, PC
```

```
1024  002320  130634                    DT1             ;$TMP1,$ERRPC
1025  002322  000000                    0
1026                              ;ITEM 101
1027  002324  125644                    EM101           ;SLU INTERRUPTS HAPPEN AT 4
1028  002326  127564                    DH1             ;TEST #, PC
1029  002330  130634                    DT1             ;$TMP1,$ERRPC
1030  002332  000000                    0
1031                              ;ITEM 102
1032  002334  125677                    EM102           ;RESET DOES NOT CLEAR XCSR<6> AND RSCR<6>
1033  002336  127564                    DH1             ;TEST #, PC
1034  002340  130634                    DT1             ;$TMP1,$ERRPC
1035  002342  000000                    0
1036                              ;ITEM 103
1037  002344  125761                    EM103           ;TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>
1038  002346  127564                    DH1             ;TEST #, PC
1039  002350  130634                    DT1             ;$TMP1,$ERRPC
1040  002352  000000                    0
1041                              ;ITEM 104
1042  002354  126034                    EM104           ;RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>
1043  002356  127564                    DH1             ;TEST #, PC
1044  002360  130634                    DT1             ;$TMP1,$ERRPC
1045  002362  000000                    0
1046                              ;ITEM 105
1047  002364  126104                    EM105           ;BREAK CONDITION DOES NOT SET RBUF PROPERLY
1048  002366  130525                    DH105           ;TEST #, PC, RBUF
1049  002370  131112                    DT105           ;$TMP1,$ERRPC,RBUF
1050  002372  000000                    0
1051                              ;ITEM 106
1052  002374  126157                    EM106           ;RBUF WASN'T CLEARED ON NEXT CHAR.
1053  002376  130525                    DH105           ;TEST #, PC, RBUF
1054  002400  131112                    DT105           ;$TMP1,$ERRPC,RBUF
1055  002402  000000                    0
1056                              ;ITEM 107
1057  002404  126235                    EM107           ;ERROR IN WRITING TO XCSR<0>
1058  002406  127564                    DH1             ;TEST #, PC
1059  002410  130634                    DT1             ;$TMP1,$ERRPC
1060  002412  000000                    0
1061                              ;ITEM 110
1062  002414  126272                    EM110           ;RESET DOES NOT CLEAR XCSR<00>
1063  002416  127564                    DH1             ;TEST #, PC
1064  002420  130634                    DT1             ;$TMP1,$ERRPC
1065  002422  000000                    0
1066                              ;ITEM 111
1067  002424  126334                    EM111           ;FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND
1068  002426  127611                    DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
1069  002430  131100                    DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
1070  002432  000000                    0
1071                              ;ITEM 112
1072  002434  126412                    EM112           ;OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF
1073  002436  130525                    DH105           ;TEST #, PC, RBUF
1074  002440  131112                    DT105           ;$TMP1,$ERRPC,RBUF
1075  002442  000000                    0
1076                              ;ITEM 113
1077  002444  126475                    EM113           ;RBUF WAS NOT CLEARED ON THE NEXT CHARACTER
1078  002446  130525                    DH105           ;TEST #, PC, RBUF
1079  002450  131112                    DT105           ;$TMP1,$ERRPC,RBUF
```

```
1080  002452  000000                          0
1081                                ;ITEM 114
1082  002454  126561                          EM114           ;ERROR IN XCSR<2>
1083  002456  127564                          DH1             ;TEST #, PC
1084  002460  130634                          DT1             ;$TMP1,$ERRPC
1085  002462  000000                          0
1086                                ;ITEM 115
1087  002464  126602                          EM115           ;ERROR IN TAG STORE FROM STANDALONE MODE
1088  002466  130557                          DH115           ;TEST #, PC, MSER, ADDRESS ACCESSED
1089  002470  131122                          DT115           ;$TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR
1090  002472  000000                          0
1091                                ;ITEM 116
1092  002474  126652                          EM116           ;DMA TAG PARITY DOES NOT SET MSER<4>
1093  002476  127564                          DH1             ;TEST #, PC
1094  002500  130634                          DT1             ;$TMP1,$ERRPC
1095  002502  000000                          0
1096                                ;ITEM 117
1097  002504  126733                          EM117           ;MSER<13>NOT SET IN STANDALONE MODE
1098  002506  127564                          DH1             ;TEST #, PC
1099  002510  130634                          DT1             ;$TMP1,$ERRPC
1100  002512  000000                          0
1101                                ;ITEM 120
1102  002514  126776                          EM120           ;DMA WRITE HITS DON'T INVALIDATE CACHE
1103  002516  127564                          DH1             ;TEST #, PC
1104  002520  130634                          DT1             ;$TMP1,$ERRPC
1105  002522  000000                          0
1106                                ;ITEM 121
1107  002524  127044                          EM121           ;IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED
1108  002526  127564                          DH1             ;TEST #, PC
1109  002530  130634                          DT1             ;$TMP1,$ERRPC
1110  002532  000000                          0
1111                                ;ITEM 122
1112  002534  127142                          EM122           ;READ DMA HIT IS MESSED UP
1113  002536  127564                          DH1             ;TEST #, PC
1114  002540  130634                          DT1             ;$TMP1,$ERRPC
1115  002542  000000                          0
1116                                ;ITEM 123
1117  002544  127170                          EM123           ;ERROR IN DMA CYCLES FROM Q22BE
1118  002546  127564                          DH1             ;TEST #, PC
1119  002550  130634                          DT1             ;$TMP1,$ERRPC
1120  002552  000000                          0
1121                                ;ITEM 124
1122  002554  127222                          EM124           ;PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS
1123  002556  127564                          DH1             ;TEST #, PC
1124  002560  130634                          DT1             ;$TMP1,$ERRPC
1125  002562  000000                          0
1126                                ;ITEM 125
1127  002564  127314                          EM125           ;NO POWER DOWN TRAP TO 24 OCCUR
1128  002566  127564                          DH1             ;TEST #, PC
1129  002570  130634                          DT1             ;$TMP1,$ERRPC
1130  002572  000000                          0
1131                                ;ITEM 126
1132  002574  127353                          EM126           ;ERROR IN INTERRUPTS FROM Q22BE
1133  002576  127564                          DH1             ;TEST #, PC
1134  002600  130634                          DT1             ;$TMP1,$ERRPC
1135  002602  000000                          0
```

L2


```
1136                                    ;ITEM 127
1137  002604  127410                            EM127           ;ERROR IN PMG COUNTER
1138  002606  127564                            DH1             ;TEST #, PC
1139  002610  130634                            DT1             ;$TMP1,$ERRPC
1140  002612  000000                            0
1141                                    ;ITEM 130
1142  002614  127452                            EM130           ;UNEXPECTED TIMEOUT
1143  002616  127564                            DH1             ;TEST #, PC
1144  002620  131136                            DT130           ;$TMP1,$ERRPC
1145  002622  000000                            0
1146                                    ;ITEM 131
1147  002624  127477                            EM131           ;ERROR WRITING TO LKS<6>
1148  002626  127564                            DH1             ;TEST #, PC
1149  002630  130634                            DT1             ;$TMP1,$ERRPC
1150  002632  000000                            0
1151                                    ;ITEM 132
1152  002634  127527                            EM132           ;MAINTENANCE REGISTER ERROR
1153  002636  127564                            DH1             ;TEST #, PC
1154  002640  130634                            DT1             ;$TMP1,$ERRPC
1155  002642  000000                            0
1156
1157                                    .SBTTL  GLOBAL VARIABLES AND REGISTER NAMES
1158
1159                                    ;REGISTERS FOR THE FIRST Q22BE
1160  002644  000000                    CSR1:   .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
1161  002646  000000                    CSR2:   .WORD   0               ;CONTROL/STATUS REGISTER 2
1162  002650  000000                    BA:     .WORD   0               ;DMA ADDRESS FOR Q22BE
1163  002652  000000                    WC:     .WORD   0               ;WORD COUNT REGISTER
1164  002654  000000                    DATA:   .WORD   0               ;DMA DATA FOR Q22BE
1165  002656  000000                    VQBE1:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
1166  002660  000000                    VQPR1:  .WORD   0               ;PRIORITY
1167  002662  000000                    SIMGOA: .WORD   0               ;SIMULTANEUOS GO ADDRESS REGISTER
1168
1169                                    ;REGISTERS FOR THE SECOND Q22BE
1170  002664  000000                    CSR12:  .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
1171  002666  000000                    CSR22:  .WORD   0               ;CONTROL/STATUS REGISTER 2
1172  002670  000000                    BA2:    .WORD   0               ;DMA ADDRESS FOR Q22BE
1173  002672  000000                    WC2:    .WORD   0               ;WORD COUNT REGISTER
1174  002674  000000                    DATA2:  .WORD   0               ;DMA DATA FOR Q22BE
1175  002676  000000                    VQBE2:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
1176  002700  000000                    VQPR2:  .WORD   0               ;PRIORITY
1177
1178  002702  000000                    LKSFL:  .WORD   0
1179  002704  000000                    ACTCHS: .WORD   0               ;ACTUAL CHECKSUM
1180  002706  000000                    SAVPCR: .WORD   0
1181  002710  000000                    SAVBR:  .WORD   0
1182          002710                    .=2710
1183  002710  000000                    TEMP:   .WORD   0
1184  002712  000017                            .BLKW   15.             ;RESERVED FOR BLOCK MODE TRANSFER
1185  002750  000000                    TIMOUT: .WORD   0
1186  002752  000032  000012  000006    Q22EN:  .WORD   32,12,6,2       ;PRIORITY 7-4 FOR Q22BE
1187  002760  000002
1188
1189
1190          177524                            BCR=            177524  ;BOOT/DIAGNOSTICS CONFIGURATION
1191          177524                            BDR=            177524  ;BOOT/DIAGNOSTICS DISPLAY
```

```
                                        BCSR=           177520          ;BOOT/DIAGNOSTICS STATUS
                                        CCR=            177746          ;CACHE CONTROL REGISTER
                                        HITMIS=         177752          ;HIT OR MISS REGISTER
                                        KMCR=           177734          ;UNIBUS CONFIGURATION REGISTER
                                        LKS=            177546          ;CLOCK STATUS REGISTER
                                        MAIREG=         177750          ;MAINTENANCE REGISTER
                                        MSER=           177744          ;MEMORY SYSTEM ERROR
                                        PCR=            177522          ;PAGE CONTROL REGISTER
                                        PIR=            177772          ;PROGRAM INTERRUPT REQUEST
                                        RBUF=           177562          ;RECEIVER DATA BUFFER
                                        RCSR=           177560          ;RECEIVER STATUS REGISTER
                                        XBUF=           177566          ;TRANSMITTER DATA BUFFER
                                        XCSR=           177564          ;TRANSMITTER STATUS REGISTER

                                        CPEREG=         177766          ;CPU ERROR REGISTER

                                        MMR0=SR0                        ;MEMORY MANAGEMENT REG. 0
                                        MMR1=SR1                        ;MEMORY MANAGEMENT REG. 1
                                        MMR2=SR2                        ;MEMORY MANAGEMENT REG. 2
                                        MMR3=SR3                        ;MEMORY MANAGEMENT REG. 3
                                        POLY= 120001
                                        NULL=           0

                                .SBTTL  GLOBAL DATA SECTION

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

                                ;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.
002762  000000                  SLOC00: .WORD   0
002764  000000                  SLOC01: .WORD   0

                                ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE WORKING DATA.
002766  000000                  LOWADD: .WORD   0               ;STORES LOW ADDRESS FOR RAM TESTS
002770  000000                  GOODAD: .WORD   0               ;STORES GOOD ADDRESS FOR RAM TESTS
002772  000000                  TSTADD: .WORD   0               ;ADDRESS STORE FOR RAM TESTS
002774  000000                  NEWADD: .WORD   0               ;ADDRESS STORE FOR RAM TEST
002776  000000                  FLAG:   .WORD   0               ;USED TO STORE "FLAG" CONDITIONS
003000  000000                  SAVSUP: .WORD   0               ;USED TO STORE SUPERVISOR STACK VALUE
003002  000000                  SAVUSE: .WORD   0               ;USED TO STORE USER STACK VALUE
003004  000000                  SAVMR0: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 0 DATA
003006  000000                  SAVMR1: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 1 DATA
003010  000000                  SAVMR2: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 2 DATA
003012  000004                  FLOAT:  .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
003022  000004                  FLO:    .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
003032  000000                  SEQ:    .WORD   0               ;STORES SEQUENCE NUMBER FOR JUMP TESTS
003034  000000                  SPS:    .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
003036  000000                  SPSJ:   .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
003040  000004                  BTEXP:  .BLKW   4               ;STORES EXPONENT DURING BIT TESTS
003050  000004                  BTRES:  .BLKW   4               ;STORES RECIEVED DATA FOR BIT TESTS
003060  000000                  COUNT:  .WORD   0               ;ERROR INDICATOR FOR FLOATING POINT TESTS
003062  000004                  RECFEC: .BLKW   4               ;RECIEVED FLOATING POINT EXCEPTION CODE
```

N2

GLOBAL DATA SECTION

SEQ 0026

```
1248  003072  000004                  RECST:  .BLKW   4               ;RECIEVED FLOATING POINT STATUS
1249  003102  000004                  RECDST: .BLKW   4               ;DESTINATION ADDRESS FOR FLOATING POINT TESTS
1250
1251                                  ;THESE LOCATIONS ARE USED BY MORE THAN ONE TEST AS LOOP COUNTERS
1252  003112  000000                  ALLCTR: .WORD   0
1253  003114  000000                  LOOPIN: .WORD   0
1254
1255                                  ;SOME MORE TEMPORARY STORAGE FOR RAM TESTS
1256  003116  000000                  SAVPOS: .WORD   0               ;STORES TEMPORARY BIT POSITIONS FOR RAM TESTS
1257  003120  000000                  MASK:   .WORD   0               ;STORES BIT MASK FOR ERROR ISOLATION
1258
1259                                  ;!!!!!THIS IS IT. THE PROGRAM TEST LOCATION!!!!!!!!!!!!!!!!!
1260  003122  000000                  TSTLOC: .WORD   0
1261  003124  000024                          .BLKW   20.
1262                                          ;FPP REGISTER DEFINITIONS
1263          000000                          AC0= %0
1264          000001                          AC1= %1
1265          000002                          AC2= %2
1266          000003                          AC3= %3
1267          000004                          AC4= %4
1268          000005                          AC5= %5
1269          000006                          AC6= %6
1270          000007                          AC7= %7
1271
1272                                          ;FPP INTERRUPT VECTOR
1273
1274          000244                          FPVEC=244
1275
1276
1277          001000                  STBOT=  1000
1278
1279
1280  003174  123456                  TAB1:   .WORD   123456
1281  003176  000000                          .WORD   000000
1282  003200  000000                          .WORD   0
1283  003202  000001                          .WORD   1
1284  003204  055555                  TAB2:   .WORD   055555
1285  003206  177777                          .WORD   177777
1286  003210  145671                          .WORD   145671
1287  003212  100000                          .WORD   100000
1288  003214  003000                  TAB3:   .WORD   003000
1289  003216  123456                          .WORD   123456
1290  003220  000000                          .WORD   0
1291  003222  000000                          .WORD   0
1292  003224  055555                  TAB4:   .WORD   55555
1293  003226  177777                          .WORD   -1
1294  003230  000000                          .WORD   0
1295  003232  000000                          .WORD   0
1296  003234  043243                  TAB5:   .WORD   43243
1297  003236  000000                          .WORD   0
1298  003240  000000                          .WORD   0
1299  003242  000000                          .WORD   0
1300  003244  162400                  TAB5A:  .WORD   162400
1301  003246  000000                          .WORD   0
1302  003250  000000                          .WORD   0
1303  003252  000000                          .WORD   0
```

```
1304  003254  000000                  TAB6:   .WORD   0
1305  003256  000000                          .WORD   0
1306  003260  000000                          .WORD   0
1307  003262  000000                          .WORD   0
1308  003264  047050                  TAB6A:  .WORD   47050
1309  003266  010000                          .WORD   10000
1310  003270  000000                          .WORD   0
1311  003272  000000                          .WORD   0
1312  003274  000200                  TAB7:   .WORD   200
1313  003276  000000                          .WORD   0
1314  003300  000000                          .WORD   0
1315  003302  000000                          .WORD   0
1316  003304  000200                  TAB8:   .WORD   200
1317  003306  000000                          .WORD   0
1318  003310  000000                          .WORD   0
1319  003312  000001                          .WORD   1
1320  003314  000400  000000  000000  TAB9:   .WORD   400.0.0.0
1321  003322  000000
1322  003324  030000                  TAB10:  .WORD   30000
1323  003326  003000                          .WORD   3000
1324  003330  000000                          .WORD   0
1325  003332  000000                          .WORD   0
1326  003334  016400                  TAB11:  .WORD   16400
1327  003336  000000                          .WORD   0
1328  003340  000000                          .WORD   0
1329  003342  000000                          .WORD   0
1330  003344  030000  003000  000002  TAB11A: .WORD   30000,3000,2,0
1331  003352  000000
1332  003354  016100  000000  000000  TAB12:  .WORD   16100,0,0,1
1333  003362  000001
1334  003364  016200                  TAB13:  .WORD   16200
1335  003366  000000                          .WORD   0
1336  003370  000000                          .WORD   0
1337  003372  000001                          .WORD   1
1338  003374  030000  003000  000000  TAB13B: .WORD   30000,3000,0,140000
1339  003402  140000
1340  003404  030000                  TAB14:  .WORD   30000
1341  003406  000000                          .WORD   0
1342  003410  000000                          .WORD   0
1343  003412  000000                          .WORD   0
1344  003414  024700                  TAB15:  .WORD   24700
1345  003416  000000                          .WORD   0
1346  003420  000000                          .WORD   0
1347  003422  000000                          .WORD   0
1348  003424  025000                  TAB16:  .WORD   25000
1349  003426  175363                          .WORD   175363
1350  003430  123456                          .WORD   123456
1351  003432  123456                          .WORD   123456
1352  003434  030000                  TAB17:  .WORD   30000
1353  003436  007020                          .WORD   7020
1354  003440  000000  000000                  .WORD   0,0
1355  003444  023456                  TAB18:  .WORD   23456
1356  003446  000000                          .WORD   0
1357  003450  000000                          .WORD   0
1358  003452  000001                          .WORD   1
1359  003454  100200  000000  000000  TAB21:  .WORD   100200,0,0,0
```

```
1360   003462  000000
1361   003464  100400  000000  000000  TAB22:  .WORD   100400,0,0,0
1362   003472  000000
1363   003474  000200  000000  000000  TAB23:  .WORD   200,0,0,1
1364   003502  000001
1365   003504  062400  000000  000000  TAB24:  .WORD   62400,0,0,0
1366   003512  000000
1367   003514  001100  000000  000000  TAB25:  .WORD   1100,0,0,0
1368   003522  000000
1369   003524  100600  000000  000000  TAB26:  .WORD   100600,0,0,0
1370   003532  000000
1371   003534  001000  000000  000000  TAB27:  .WORD   1000,0,0,0
1372   003542  000000
1373   003544  000600  000000  000000  TAB28:  .WORD   600,0,0,0
1374   003552  000000
1375   003554  010100  000000  000000  TAB29:  .WORD   10100,0,0,0
1376   003562  000000
1377   003564  010100  000000  002000  TAB29A: .WORD   10100,0,2000,0
1378   003572  000000
1379                                   ;
1380   003574  000500  000000  000000  TAB30:  .WORD   500,0,0,0
1381   003602  000000
1382   003604  100400  000000  000000  TAB31:  .WORD   100400,0,0,0
1383   003612  000000
1384   003614  016000  000000  000000  TAB32:  .WORD   16000,0,0,0
1385   003622  000000
1386   003624  011600  000000  000000  TAB33:  .WORD   11600,0,0,0
1387   003632  000000
1388   003634  000640  000000  000000  TAB34:  .WORD   640,0,0,0
1389   003642  000000
1390   003644  077600  000000  000000  TAB40:  .WORD   77600,0,0,0
1391   003652  000000
1392   003654  100200  000000  000000  TAB41:  .WORD   100200,0,0,1
1393   003662  000001
1394   003664  000340  000000  000000  TAB42:  .WORD   340,0,0,0
1395   003672  000000
1396   003674  000077  177777  177777  TAB43:  .WORD   77,177777,177777,177776
1397   003702  177776
1398   003704  000577  177777  177777  TAB45:  .WORD   577,-1,-1,-1
1399   003712  177777
1400   003714  000577  177777  000000  TAB46:  .WORD   577,-1,0,0
1401   003722  000000
1402   003724  173737  124242  052525  TAB47:  .WORD   173737,124242,052525,12346
1403   003732  012346
1404   003734  000000  000000  052525  TAB47A: .WORD   0,0,052525,12346
1405   003742  012346
1406   003744  173737  124242  000000  TAB48:  .WORD   173737,124242,0,0
1407   003752  000000
1408   003754  000600  000000  000000  TAB49:  .WORD   600,0,0,0
1409   003762  000000
1410
1411   003764                          START:
1412                                   ;; LCP/ORION ROUTINE TO SAVE EMTULATOR AND PRIORITY
1413
1414   003764  005737  004052          EMTSAV: TST     SAV30           ;; FIRST TIME THROUGH ?
1415   003770  001034                          BNE     VMKOR           ;; BRANCH IF BEEN HERE ALREADY
```

D3


```
1416  003772  032737  000040  000052          BIT     #BIT5,@#52              ;; ARE WE IN UFD MODE ?
1417  004000  001430                          BEQ     VMKOR                   ;; LEAVE IF NOT
1418  004002  012737  177777  004056          MOV     #-1,UFDFLG              ;; SET UFD FLAG
1419  004010  032737  000100  000052          BIT     #BIT6,@#52              ;; ARE WE IN QUIET MODE ?
1420  004016  001403                          BEQ     1$                      ;; BR IF NOT
1421  004020  012737  177777  004060          MOV     #-1,UQUIET              ;; SET QUIET MODE
1422  004026  104042                  1$:     EMT     42                      ;; GET ADDRESS OF XXDP DCA TABLE
1423  004030  005060  000042                  CLR     42(R0)                  ;; CLR XXDP+ "DRSERR"
1424  004034  013737  000030  004052          MOV     30,SAV30                ;; SAVE EMULATOR ADDRESS
1425  004042  013737  000032  004054          MOV     32,SAV32                ;; SAVE EMULATOR PRIORITY LEVEL
1426  004050  000404                          BR      VMKOR                   ;; GET AROUND TAG AREA
1427  004052  000000                  SAV30:  .WORD   0                       ;; PUT EMULATOR INFO HERE
1428  004054  00000C                  SAV32:  .WORD   0                       ;; PUT PRIORITY LOCATION        HERE
1429  004056  000000                  UFDFLG: .WORD   0                       ;; USER FRIENDLY MODE FLAG
1430  004060  000000                  UQUIET: .WORD   0                       ;; UFD QUIET MODE FLAG
1431  004062                          VMKOR:
1432                                  ;*****************************************************************************
1433
1434
1435  004062                          1$:
1436                                  .SBTTL  INITIALIZE THE COMMON TAGS
1437                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1438  004062  012706  001100                  MOV     #$CMTAG,R6              ;;FIRST LOCATION TO BE CLEARED
1439  004066  005026                          CLR     (R6)+                   ;;CLEAR MEMORY LOCATION
1440  004070  022706  001140                  CMP     #SWR,R6 ;;DONE?
1441  004074  001374                          BNE     .-6                     ;;LOOP BACK IF NO
1442  004076  012706  001100                  MOV     #STACK,SP               ;;SETUP THE STACK POINTER
1443                                  ;;INITIALIZE A FEW VECTORS
1444  004102  012737  133054  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1445  004110  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1446  004116  012737  133342  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1447  004124  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1448  004132  012737  136060  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1449  004140  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
1450  004146  012737  136142  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1451  004154  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1452  004162  013737  132766  132760          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
1453  004170  005037  001164                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1454  004174  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1455  004200  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1456  004206  012737  004206  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1457  004214  012737  004214  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1458                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1459                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1460  004222  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1461  004226  012737  004262  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1462  004234  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1463  004242  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1464  004250  022777  177777  174662          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1465  004256  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1466                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1467  004260  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1468  004262  012716  004270          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1469  004266  000002                          RTI
1470  004270  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1471  004276  012737  000174  001142          MOV     #DISPREG,DISPLAY
```

E 4

INITIALIZE THE COMMON TAGS

SEQ 0030

```
1472  004304  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1473
1474  004310  005037  001206                  CLR     $PASS           ;;CLEAR PASS COUNT
1475  004314  132737  000200  001221          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1476  004322  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
1477  004324  012737  001222  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1478  004332                          67$:
1479  004332  012737  132334  000004          MOV     #TOUT,@#ERRVEC  ;POINT TO TIMEOUT ROUTINE
1480  004340  012737  000340  000006          MOV     #340,@#ERRVEC+2 ;AT PRIORITY 7
1481  004346  012737  131600  000114          MOV     #RAMPAR,@#114   ;POINT PARITY ABORT
1482  004354  012737  000340  000116          MOV     #340,@#116      ;AT PRIORITY7
1483  004362  012737  132516  000250          MOV     #MMUTRP,@#250   ;POINT MMU TRAP VECTOR
1484  004370  012737  000340  000252          MOV     #340,@#252      ;
1485  004376  005037  177766                  CLR     @#177766        ;CLEAR CPU ERROR REGISTER
1486  004402  032737  000040  000052          BIT     #BIT05,@#52     ;IN UFD MODE ?
1487  004410  001051                          BNE     LOOP            ;IF SO, SKIP PRINTOUT
1488                                  .SBTTL  TYPE PROGRAM NAME
1489                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1490  004412  005227  177777                  INC     #-1             ;;FIRST TIME?
1491  004416  001044                          BNE     68$             ;;BRANCH IF NO
1492  004420  022737  133020  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
1493  004426  001440                          BEQ     68$             ;;BRANCH IF YES
1494  004430  104401  004476                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1495                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1496  004434  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1497  004440  001012                          BNE     70$             ;;BRANCH IF YES
1498  004442  123727  001220  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1499  004450  001406                          BEQ     70$             ;;BRANCH IF YES
1500  004452  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1501  004460  001005                          BNE     71$             ;;BRANCH IF NO
1502  004462  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1503  004464  000403                          BR      71$
1504  004466  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1505  004474                          71$:
1506  004474  000415                          BR      68$             ;;GET OVER THE ASCIZ
1507                                  ;;69$:  .ASCIZ  <CRLF><KDJ11-B CPU DIAGNOSTIC><CRLF>
1508  004530                          68$:
1509  004530  004737  131656                  JSR     PC,Q22SIZ       ;SIZE FOR Q22BE
1510  004534                          LOOP:
1511                                  .DSABLE AMA
1512                                  ;;*****************************************************************
1513  004534  000004                  TST1:   SCOPE
1514                                  .SBTTL  BASE INSTRUCTION SET TESTS
1515                                  ;*************************************************************************
1516                                  ;*************************************************************************
1517                                  ;                       BEGIN BASE INSTRUCTION SET TESTING
1518                                  ;*************************************************************************
1519                                  ;*************************************************************************
1520  004536                          FRSTST:
1521
1522                                  ;TEST BEQ BNE INSTRUCTIONS
1523                                  ;;*************************************************************
1524                                  ;THESE TWO INSTRUCTIONS ARE FUNDAMENTAL TO RECOGNIZING ERROR CONDITIONS
1525  004536  000277                          SCC
1526  004540  000244                          CLZ                     ;CC=0100 - Z BIT CLEARED
1527  004542  001401                          BEQ     1$              ;*TEST INSTR -TRY TO CAUSE A BEQ ERROR
```

```
1528  004544  001001                          BNE     2$              ;BRANCH IF GOOD
1529                                                                  ;THE Z FLAG DIDNT CLEAR OR BRANCH FAILED.
1530                                                                  ;FAILURE AT THIS LOCATION
1531                                                                  ;COULD MEAN A BUS PROBLEM, MICRO-CODE PROBLEM
1532                                                                  ;CONDITION CODE PROBLEM OR JUST ABOUT ANYTHING
1533                                                                  ;ELSE.
1534  004546  104001                    1$:   ERROR   ;1              ;CPU ERROR
1535  004550  000257                    2$:   CCC                     ;
1536  004552  000264                          SEZ                     ;COND CODES = 0100 (ZERO)
1537  004554  001001                          BNE     3$              ;*TEST INSTR* TRY TO BRANCH ON ZERO FLAG
1538  004556  001401                          BEQ     4$              ;*TEST INSTR* BRANCH IF GOOD
1539                                                                  ;BRANCH FAILURE WITH Z BIT SET
1540  004560  104001                    3$:   ERROR   ;1              ;CPU ERROR
1541  004562                             4$:                          ;END OF TEST
1542  004562                             M2:
1543
1544                                     ;       TEST BRANCH ON CARRY
1545                                     ;;**************************************************************
1546                                     ;THIS IS A TEST TO SEE IF THE MODULE FORM ANTICIPATED IS FEASIBLE.
1547  004562  000257                          CCC                     ;CC=0000
1548  004564  103001                          BCC     2$              ;*TEST INSTR*
1549                                                                  ;BRANCH CARRY CLEAR FAILED
1550  004566  104001                    1$:   ERROR   ;1              ;CPU ERROR
1551  004570  000261                    2$:   SEC                     ;CC=1111
1552  004572  103401                          BCS     4$              ;*TEST INSTR*
1553                                                                  ; BRANCH CARRY SET FAILED
1554  004574  104001                    3$:   ERROR   ;1              ;CPU ERROR
1555  004576                             4$:
1556
1557  004576                             M3:
1558
1559                                     ;       TEST DATA PATHS
1560  004576  005000                          CLR     R0
1561  004600  001005                          BNE     1$              ;TRY TO INSURE WE ARE TESTING
1562                                                                  ;THE DATA PATH AND NOT THE "CLR R0" INSTRUCTION
1563  004602  005010                          CLR     (R0)            ;FORCE LOCATION TO ZERO
1564  004604  001003                          BNE     1$              ;TRY TO INSURE 0=0
1565  004606  005737  000000                  TST     @#0             ;AGAIN, TRY TO INSURE THAT 0=0
1566  004612  001401                          BEQ     2$              ;BRANCH IF GOOD
1567                                                                  ;LOCATION 0 NOT SETUP PROPERLY
1568  004614  104001                    1$:   ERROR   ;1              ;CPU ERROR
1569  004616                             2$:
1570  004616                             M4:
1571
1572                                     ;       TEST DATA PATHS - ONES AND ZEROS
1573  004616  012737  125252  000000          MOV     #125252,@#0             ;0=125252
1574  004624  022737  125252  000000          CMP     #125252,@#0             ;SEE IF DATA MADE IT
1575  004632  001401                          BEQ     2$                      ;BRANCH IF IF DATA IS GOOD
1576                                                                          ;ERROR! EITHER THE BUS IS BAD,
1577                                                                          ;OR THE MOV OR COMPARE
1578                                                                          ;INSTRUCTIONS FAILED
1579  004634  104001                    1$:   ERROR   ;1              ;CPU ERROR
1580  004636                             2$:                          ;END OF TEST
1581
1582                                     ;
1583  004636                             M5:
```

```
1584
1585                                    ;       TEST DATA PATHS - DATA 0'S AND 1'S
1586  004636  012737  052525  000000            MOV     #052525,@#0             ;SETUP DATA
1587  004644  023727  000000  052525            CMP     @#0,#052525             ; TEST FOR CORRECT DATA
1588  004652  001401                            BEQ     2$
1589  004654  104001                    1$:     ERROR   +1              ;CPU ERROR
1590  004656                            2$:
1591                                    ;
1592  004656                            M6:
1593                                    ;       TEST DATA PATHS - 1'S
1594  004656  005037  000000                    CLR     @#0
1595  004662  005137  000000                    COM     @#0             ;SET UP MEMORY LOCATION 0 - 111111
1596  004666  023727  000000  177777            CMP     @#0,#177777     ; TEST DATA
1597  004674  001401                            BEQ     2$              ;BRANCH IF NO ERROR
1598  004676  104001                    1$:     ERROR   +1              ;CPU ERROR
1599  004700                            2$:
1600
1601                                    ;
1602  004700                            GPROTS:
1603                                    ;       R0 BIT TESTS
1604  004700  012700  177777                    MOV     #177777,R0              ;R0=177777
1605  004704  020027  177777                    CMP     R0,#177777              ;DOES R0=177777
1606  004710  001401                            BEQ     1$                      ;YES GO ON
1607                                                                            ;NO GO TO ERROR
1608  004712  104001                            ERROR   +1              ;CPU ERROR
1609  004714  005000                    1$:     CLR     R0                      ;R0=0
1610  004716  020027  000000                    CMP     R0,#0                   ;DOES R0=0
1611  004722  001401                            BEQ     2$                      ;YES GO ON
1612                                                                            ;NO GO TO ERROR
1613  004724  104001                            ERROR   +1              ;CPU ERROR
1614  004726  012700  125252            2$:     MOV     #125252,R0              ;R0=125252
1615  004732  020027  125252                    CMP     R0,#125252              ;DOES R0=125252
1616  004736  001401                            BEQ     3$                      ;YES GO ON
1617                                                                            ;NO GO TO ERROR
1618  004740  104001                            ERROR   +1              ;CPU ERROR
1619  004742  012700  052525            3$:     MOV     #52525,R0               ;R0=52525
1620  004746  020027  052525                    CMP     R0,#52525               ;DOES R0=52525
1621  004752  001401                            BEQ     4$                      ;YES GO ON
1622                                                                            ;NO GO TO ERROR
1623  004754  104001                            ERROR   +1              ;CPU ERROR
1624  004756                            4$:
1625                                    ;
1626  004756                            GPR1TS:
1627                                    ;       R1 BIT TESTS
1628  004756  012701  177777                    MOV     #177777,R1              ;R1=177777
1629  004762  020127  177777                    CMP     R1,#177777              ;DOES R1=177777
1630  004766  001401                            BEQ     1$                      ;YES GO ON
1631                                                                            ;NO GO TO ERROR
1632  004770  104001                            ERROR   +1              ;CPU ERROR
1633  004772  005001                    1$:     CLR     R1                      ;R1=0
1634  004774  020127  000000                    CMP     R1,#0                   ;DOES R1=0
1635  005000  001401                            BEQ     2$                      ;YES GO ON
1636                                                                            ;NO GO TO ERROR
1637  005002  104001                            ERROR   +1              ;CPU ERROR
1638  005004  012701  125252            2$:     MOV     #125252,R1              ;R1=125252
1639  005010  020127  125252                    CMP     R1,#125252              ;DOES R1=125252
```

```
1640  005014  001401                          BEQ     3$                      ;YES GO ON
1641                                                                          ;NO GO TO ERROR
1642  005016  104001                          ERROR   +1              ;CPU ERROR
1643  005020  012701  052525          3$:     MOV     #52525,R1               ;R1=52525
1644  005024  020127  052525                  CMP     R1,#52525               ;DOES R1=52525
1645  005030  001401                          BEQ     4$                      ;YES GO ON
1646                                                                          ;NO GO TO ERROR
1647  005032  104001                          ERROR   +1              ;CPU ERROR
1648  005034                          4$:
1649                                          ;
1650  005034                          GPR2TS:
1651                                          ;       R2 BIT TESTS
1652  005034  012702  177777                  MOV     #177777,R2              ;R2=177777
1653  005040  020227  177777                  CMP     R2,#177777              ;DOES R2=177777
1654  005044  001401                          BEQ     1$                      ;YES GO ON
1655                                                                          ;NO GO TO ERROR
1656  005046  104001                          ERROR   +1              ;CPU ERROR
1657  005050  005002                  1$:     CLR     R2                      ;R2=0
1658  005052  020227  000000                  CMP     R2,#0                   ;DOES R2=0
1659  005056  001401                          BEQ     2$                      ;YES GO ON
1660                                                                          ;NO GO TO ERROR
1661  005060  104001                          ERROR   +1              ;CPU ERROR
1662  005062  012702  125252          2$:     MOV     #125252,R2              ;R2=125252
1663  005066  020227  125252                  CMP     R2,#125252              ;DOES R2=125252
1664  005072  001401                          BEQ     3$                      ;YES GO ON
1665                                                                          ;NO GO TO ERROR
1666  005074  104001                          ERROR   +1              ;CPU ERROR
1667  005076  012702  052525          3$:     MOV     #52525,R2               ;R2=52525
1668  005102  020227  052525                  CMP     R2,#52525               ;DOES R2=52525
1669  005106  001401                          BEQ     4$                      ;YES GO ON
1670                                                                          ;NO GO TO ERROR
1671  005110  104001                          ERROR   +1              ;CPU ERROR
1672  005112                          4$:
1673                                          ;
1674  005112                          GPR3TS:
1675                                          ;       R3 BIT TESTS
1676  005112  012703  177777                  MOV     #177777,R3              ;R3=177777
1677  005116  020327  177777                  CMP     R3,#177777              ;DOES R3=177777
1678  005122  001401                          BEQ     1$                      ;YES GO ON
1679                                                                          ;NO GO TO ERROR
1680  005124  104001                          ERROR   +1              ;CPU ERROR
1681  005126  005003                  1$:     CLR     R3                      ;R3=0
1682  005130  020327  000000                  CMP     R3,#0                   ;DOES R3=0
1683  005134  001401                          BEQ     2$                      ;YES GO ON
1684                                                                          ;NO GO TO ERROR
1685  005136  104001                          ERROR   +1              ;CPU ERROR
1686  005140  012703  125252          2$:     MOV     #125252,R3              ;R3=125252
1687  005144  020327  125252                  CMP     R3,#125252              ;DOES R3=125252
1688  005150  001401                          BEQ     3$                      ;YES GO ON
1689                                                                          ;NO GO TO ERROR
1690  005152  104001                          ERROR   +1              ;CPU ERROR
1691  005154  012703  052525          3$:     MOV     #52525,R3               ;R3=52525
1692  005160  020327  052525                  CMP     R3,#52525               ;DOES R3=52525
1693  005164  001401                          BEQ     4$                      ;YES GO ON
1694                                                                          ;NO GO TO ERROR
1695  005166  104001                          ERROR   +1              ;CPU ERROR
```

```
1696  005170                          4$:
1697                                  ;
1698  005170                          GPR4TS:
1699                                  ;       R4 BIT TESTS
1700  005170  012704  177777                 MOV     #177777,R4              ;R4=177777
1701  005174  020427  177777                 CMP     R4,#177777              ;DOES R4=177777
1702  005200  001401                         BEQ     1$                      ;YES GO ON
1703                                                                          ;NO GO TO ERROR
1704  005202  104001                         ERROR   .+1             ;CPU ERROR
1705  005204  005004                  1$:    CLR     R4                      ;R4=0
1706  005206  020427  000000                 CMP     R4,#0                   ;DOES R4=0
1707  005212  001401                         BEQ     2$                      ;YES GO ON
1708                                                                          ;NO GO TO ERROR
1709  005214  104001                         ERROR   .+1             ;CPU ERROR
1710  005216  012704  125252          2$:    MOV     #125252,R4              ;R4=125252
1711  005222  020427  125252                 CMP     R4,#125252              ;DOES R4=125252
1712  005226  001401                         BEQ     3$                      ;YES GO ON
1713                                                                          ;NO GO TO ERROR
1714  005230  104001                         ERROR   .+1             ;CPU ERROR
1715  005232  012704  052525          3$:    MOV     #052525,R4              ;R4=52525
1716  005236  020427  052525                 CMP     R4,#052525              ;DOES R4=52525
1717  005242  001401                         BEQ     4$                      ;YES GO ON
1718                                                                          ;NO GO TO ERROR
1719  005244  104001                         ERROR   .+1             ;CPU ERROR
1720  005246                          4$:
1721                                  ;
1722  005246                          GPR5TS:
1723                                  ;       R5 BIT TESTS
1724  005246  012705  177777                 MOV     #177777,R5              ;R5=177777
1725  005252  020527  177777                 CMP     R5,#177777              ;DOES R5=177777
1726  005256  001401                         BEQ     1$                      ;YES GO ON
1727                                                                          ;NO GO TO ERROR
1728  005260  104001                         ERROR   .+1             ;CPU ERROR
1729  005262  005005                  1$:    CLR     R5                      ;R5=0
1730  005264  020527  000000                 CMP     R5,#0                   ;DOES R5=0
1731  005270  001401                         BEQ     2$                      ;YES GO ON
1732                                                                          ;NO GO TO ERROR
1733  005272  104001                         ERROR   .+1             ;CPU ERROR
1734  005274  012705  125252          2$:    MOV     #125252,R5              ;R5=125252
1735  005300  020527  125252                 CMP     R5,#125252              ;DOES R5=125252
1736  005304  001401                         BEQ     3$                      ;YES GO ON
1737                                                                          ;NO GO TO ERROR
1738  005306  104001                         ERROR   .+1             ;CPU ERROR
1739  005310  012705  052525          3$:    MOV     #052525,R5              ;R5=52525
1740  005314  020527  052525                 CMP     R5,#052525              ;DOES R5=52525
1741  005320  001401                         BEQ     4$                      ;YES GO ON
1742                                                                          ;NO GO TO ERROR
1743  005322  104001                         ERROR   .+1             ;CPU ERROR
1744  005324                          4$:
1745                                  ;
1746  005324                          GPR6TS:
1747                                  ;       R6 BIT TESTS
1748  005324  012706  177777                 MOV     #177777,R6              ;R6=177777
1749  005330  020627  177777                 CMP     R6,#177777              ;DOES R6=177777
1750  005334  001401                         BEQ     1$                      ;YES GO ON
1751                                                                          ;NO GO TO ERROR
```

J3


```
1752  005336  104001                        ERROR   ;1              ;CPU ERROR
1753  005340  005006                1$:     CLR     R6                      ;R6=0
1754  005342  020627  000000                CMP     R6,#0                   ;DOES R6=0
1755  005346  001401                        BEQ     2$                      ;YES GO ON
1756                                                                        ;NO GO TO ERROR
1757  005350  104001                        ERROR   ;1              ;CPU ERROR
1758  005352  012706  125252        2$:     MOV     #125252,R6              ;R6=125252
1759  005356  020627  125252                CMP     R6,#125252              ;DOES R6=125252
1760  005362  001401                        BEQ     3$                      ;YES GO ON
1761                                                                        ;NO GO TO ERROR
1762  005364  104001                        ERROR   ;1              ;CPU ERROR
1763  005366  012706  052525        3$:     MOV     #52525,R6               ;R6=52525
1764  005372  020627  052525                CMP     R6,#52525               ;DOES R6=52525
1765  005376  001401                        BEQ     4$                      ;YES GO ON
1766                                                                        ;NO GO TO ERROR
1767  005400  104001                        ERROR   ;1              ;CPU ERROR
1768  005402  012706  001000        4$:     MOV     #STBOT,R6               ;RESTORE SP
1769
1770                                        ;
1771  005406                        PSWBTS:
1772                                ;       PSW LOW BYTE BIT TESTS
1773  005406  012737  000377  177776        MOV     #377,@#177776           ;PS=357 T BIT SHOULDN'T SET
1774  005414  022737  000357  177776        CMP     #357,@#177776           ;DOES PS=357
1775  005422  001401                        BEQ     1$                      ;YES GO ON
1776                                                                        ;NO GO TO ERROR
1777  005424  104001                        ERROR   ;1              ;CPU ERROR
1778  005426  005037  177776        1$:     CLR     @#177776                ;PS=0
1779  005432  022737  000000  177776        CMP     #0,@#177776             ;DOES PS=0
1780  005440  001401                        BEQ     2$                      ;YES GO ON
1781                                                                        ;NO GO TO ERROR
1782  005442  104001                        ERROR   ;1              ;CPU ERROR
1783  005444  012737  000105  177776  2$:   MOV     #105,@#177776           ;PS=105
1784  005452  022737  000105  177776        CMP     #105,@#177776           ;DOES PS=105
1785  005460  001401                        BEQ     3$                      ;YES GO ON
1786                                                                        ;NO GO TO ERROR
1787  005462  104001                        ERROR   ;1              ;CPU ERROR
1788  005464  012737  000252  177776  3$:   MOV     #252,@#177776           ;PS=252
1789  005472  022737  000252  177776        CMP     #252,@#177776           ;DOES PS=252
1790  005500  001401                        BEQ     4$                      ;YES GO ON
1791                                                                        ;NO GO TO ERROR
1792  005502  104001                        ERROR   ;1              ;CPU ERROR
1793  005504                        4$:
1794
1795  005504                        MSPO:
1796
1797                                ;       TEST SINGLE OPERAND INSTRUCTIONS- MODE 0
1798                                ;;*****************************************************************
1799                                ;THE INC, COM, CLR, AND DECREMENT INSTRUCTIONS ARE VERIFED.
1800  005504  005004                        CLR     R4                      ;INITIALIZE R4 WITH DATA
1801  005506  005104                        COM     R4                      ;
1802  005510  005004                        CLR     R4                      ;*TEST INSTRUCTION
1803  005512  001401                        BEQ     2$                      ;BRANCH IF R4 CLEARED
1804  005514  104001                1$:     ERROR   ;1              ;CPU ERROR
1805  005516  005104                2$:     COM     R4                      ;*TEST COMPLIMENT INSTRUCTION
1806  005520  005204                        INC     R4                      ;*TEST INCREMENT INSTRUCTION
1807  005522  001401                        BEQ     4$                      ;BRANCH IF R4 =0
```

```
                                                                                ;COMPLIMENT OR INCREMENT FAILED
005524  104001                  3$:     ERROR   +1              ;CPU ERROR
005526                          4$:
                                ;
005526                          MSPB:
                                ;       TEST SINGLE OPS - EVEN BYTE OF CLRB, DECB, AND COMB
005526  005004                          CLR     R4
005530  005104                          COM     R4              ;SETUP TEST REGISTER
005532  105004                          CLRB    R4              ;*TEST CLEAR BYTE INSTRUCTION
005534  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;CLEAR EVEN BYTE FAILED
005536  104001                  1$:     ERROR   +1              ;CPU ERROR
005540  105304                  2$:     DECB    R4              ;*TEST DECREMENT BYTE
005542  100002                          BPL     3$              ;DECREMENT BYTE FAILED
005544  105104                          COMB    R4              ;*TEST COMPLIMENT BYTE
005546  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;COMPLIMENT OR DECREMENT FAILED TO WORK
005550  104001                  3$:     ERROR   +1              ;CPU ERROR
005552                          4$:
                                ;
005552                          MSPC:
                                ;       TEST SINGLE OPS - MODE 1 CLRB, COMB, AND INCB
005552  005004                          CLR     R4
005554  005014                          CLR     (R4)
005556  005114                          COM     (R4)            ;SETUP TEST DATA
005560  005014                          CLR     (R4)            ;*TEST INSTRUCTION
005562  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;MODE 1 FAILED
005564  104001                  1$:     ERROR   +1              ;CPU ERROR
005566  005114                  2$:     COM     (R4)            ;*TEST INSTRUCTION
005570  001403                          BEQ     3$              ;(0)SHOULD = -1
005572  100002                          BPL     3$              ;
005574  005214                          INC     (R4)            ;*TEST INSTRUCTION
005576  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;COM OR INC FAILED TO ALTER LOC 0 CORRECTLY
005600  104001                  3$:     ERROR   +1              ;CPU ERROR
005602                          4$:
                                ;
005602                          MSPD:
                                ;       TEST SINGLE OPS MODE1-EVEN BYTE-CLRB,COMB,INCB
005602  005004                          CLR     R4
005604  005014                          CLR     (R4)
005606  005114                          COM     (R4)            ;SETUP TEST DATA
005610  105014                          CLRB    (R4)            ;*TEST INSTRUCTION
005612  105014                          CLRB    (R4)            ;*TEST INSTRUCTION
005614  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;CLEAR (0) EVEN BYTE FAILED
005616  104001                  1$:     ERROR   +1              ;CPU ERROR
005620  105214                  2$:     INCB    (R4)            ;*TEST INSTRUCTION
005622  100405                          BMI     3$              ; TEST FLAGS
005624  001404                          BEQ     3$
```

L3


```
1864  005626  105114                  COMB    (R4)            ;*TEST INSTRUCTION
1865  005630  105214                  INCB    (R4)
1866  005632  105214                  INCB    (R4)
1867  005634  001401                  BEQ     4$              ;BRANCH IF GOOD
1868                                                          ;COMB OR INCB FAILED
1869  005636  104001          3$:     ERROR   +1              ;CPU ERROR
1870  005640                  4$:
1871
1872                          ;
1873  005640                  MSPE0:
1874
1875                          ;       TEST SINGLE OPS - ODD BYTE - CLRB, COMB, DECB
1876  005640  005004                  CLR     R4
1877  005642  005014                  CLR     (R4)
1878  005644  005114                  COM     (R4)            ;SETUP TEST DATA
1879  005646  005204                  INC     R4              ;POINT TO ODD BYTE
1880  005650  105014                  CLRB    (R4)            ;*TEST INSTRUCTION
1881  005652  001401                  BEQ     1$              ;BRANCH IF GOOD
1882                                                          ;CLEAR ODD BYTE FAILED
1883  005654  104001                  ERROR   +1              ;CPU ERROR
1884  005656  005304          1$:     DEC     R4              ;POINT TO EVEN BYTE
1885  005660  005214                  INC     (R4)            ;LOC 0=1 0
1886  005662  005204                  INC     R4              ;POINT TO ODD BYTE
1887  005664  105114                  COMB    (R4)            ;*TEST INSTRUCTION
1888  005666  105214                  INCB    (R4)            ;LOC 0=-1 0
1889  005670  100003                  BPL     2$              ;BRANCH IF ERROR
1890  005672  001402                  BEQ     2$              ;
1891  005674  105214                  INCB    (R4)            ;*TEST INSTRUCTION
1892  005676  001401                  BEQ     3$              ;BRANCH IF GOOD
1893                                                          ;MODE 1, ODD BYTE FAILED
1894  005700  104001          2$:     ERROR   +1              ;CPU ERROR
1895  005702                  3$:
1896
1897                          ;
1898  005702                  MSPF:
1899                          ;       TEST SINGLE OP - MODE 2 - CLR, COM, INC
1900  005702  005004                  CLR     R4
1901  005704  105104                  COMB    R4
1902  005706  005204                  INC     R4              ;R4=400
1903  005710  005014                  CLR     (R4)            ;400=0
1904  005712  005114                  COM     (R4)            ;400=-1
1905  005714  005024                  CLR     (R4)+           ;*TEST INSTRUCTION
1906  005716  001401                  BEQ     1$              ;BRANCH IF GOOD
1907                                                          ;MODE 2 CLEAR FAILED
1908  005720  104001                  ERROR   +1              ;CPU ERROR
1909  005722  005304          1$:     DEC     R4
1910  005724  005304                  DEC     R4              ;R4=400
1911  005726  005124                  COM     (R4)+           ;*TEST INSTRUCTION
1912  005730  100004                  BPL     2$              ;BRANCH IF FAILURE
1913  005732  005304                  DEC     R4
1914  005734  005304                  DEC     R4              ;R4=400
1915  005736  005224                  INC     (R4)+           ;*TEST INSTRUCTION
1916  005740  001401                  BEQ     3$              ;BRANCH IF GOOD
1917                                                          ;MODE 2 FAILURE
1918  005742  104001          2$:     ERROR   +1              ;CPU ERROR
1919  005744                  3$:
```

```
                                        ;
                                MSPG:
                                        ;       TEST CLRB, COMB, DECB, MODE 2 - EVEN BYTE
                                        ;
005744  005004                          CLR     R4
005746  105104                          COMB    R4
005750  005204                          INC     R4              ;R4=400
005752  005014                          CLR     (R4)
005754  005114                          COM     (R4)            ;400=-1
005756  105024                          CLRB    (R4)+           ;*TEST INSTRUCTION
005760  001401                          BEQ     1$              ;BRANCH IF GOOD
                                                                ;MODE 2 EVEN BYTE FAILED
005762  104001                          ERROR   +1              ;CPU ERROR
005764  005304                  1$:     DEC     R4
005766  105324                          DECB    (R4)+           ;*TEST INSTRUCTION
005770  100003                          BPL     2$              ;BRANCH IF BAD
005772  005304                          DEC     R4              ;POINT TO EVEN BYTE
005774  105124                          COMB    (R4)+           ;*TEST INSTRUCTION
005776  001401                          BEQ     3$              ;BRANCH IF GOOD
                                                                ;MODE 2, EVEN BYTE FAILED
006000  104001                  2$:     ERROR   +1              ;CPU ERROR
006002                          3$:
                                        ;
                                MSPH:
                                        ;       TEST CLRB, COMB, INCB MODE 2 - ODD BYTE
                                        ;
006002  005004                          CLR     R4
006004  105104                          COMB    R4
006006  005204                          INC     R4              ;R4=400
006010  005014                          CLR     (R4)
006012  005114                          COM     (R4)            ;400=-1 -1
006014  005214                          INC     (R4)            ;POINT TO ODD BYTE
006016  105024                          CLRB    (R4)+           ;*TEST INSTRUCTION
006020  001401                          BEQ     1$              ;BRANCH IF GOOD
                                                                ;MODE 2,ODD BYTE FAILED
006022  104001                          ERROR   +1              ;CPU ERROR
006024  005304                  1$:     DEC     R4              ;POINT TO ODD BYTE
006026  005304                          DEC     R4
006030  105224                          INCB    (R4)+           ;400=1 0
006032  105124                          COMB    (R4)+           ;*TEST INSTRUCTION
006034  100003                          BPL     2$              ;BRANCH IF MODE 2 FAILED
006036  005304                          DEC     R4              ;POINT TO ODD BYTE
006040  105224                          INCB    (R4)+
006042  001401                          BEQ     3$              ;BRANCH IF GOOD
                                                                ;MODE 2,ODD BYTE FAILED
006044  104001                  2$:     ERROR   +1              ;CPU ERROR
006046                          3$:
                                        ;
                                MSPI:
                                        ;       TEST CLR, COM, INC - MODE 3
                                        ;
006046  005004                          CLR     R4              ;
006050  005014                          CLR     (R4)            ;0=0
```

```
 1976  006052  105114                  COMB    (R4)            ;
 1977  006054  005214                  INC     (R4)            ;0=400
 1978  006056  005034                  CLR     @(R4)+          ;*TEST INSTRUCTION
 1979  006060  001401                  BEQ     1$              ;BRANCH IF GOOD
 1980                                                          ;MODE 3 FAILED, 400 SHOULD=0
 1981  006062  104001                  ERROR   +1              ;CPU ERROR
 1982  006064  005304          1$:     DEC     R4
 1983  006066  005304                  DEC     R4              ;R4=0
 1984  006070  005134                  COM     @(R4)+          ;*TEST INSTRUCTION
 1985  006072  100004                  BPL     2$              ;BRANCH IF BAD
 1986  006074  005304                  DEC     R4
 1987  006076  005304                  DEC     R4              ;REPOSITION POINTER
 1988  006100  005234                  INC     @(R4)+          ;*TEST INSTRUCTION
 1989  006102  001401                  BEQ     3$              ;BRANCH IF GOOD
 1990                                                          ;MODE 3 FAILED
 1991  006104  104001          2$:     ERROR   +1              ;CPU ERROR
 1992  006106                  3$:
 1993
 1994                                  ;
 1995  006106                  MSPJ:
 1996
 1997                                  ;       TEST CLRB, COMB, INCB - MODE 3, EVEN/ODD BYTE
 1998  006106  005004                          CLR     R4              ;R4=0
 1999  006110  005001                          CLR     R1
 2000  006112  105101                          COMB    R1
 2001  006114  005201                          INC     R1              ;R1=400
 2002  006116  005011                          CLR     (R1)
 2003  006120  005121                          COM     (R1)+           ;400=-1
 2004  006122  005011                          CLR     (R1)            ;
 2005  006124  105111                          COMB    (R1)            ;402=000 377
 2006  006126  005014                          CLR     (R4)            ;
 2007  006130  105114                          COMB    (R4)            ;
 2008  006132  005214                          INC     (R4)            ;0=400
 2009  006134  105034                          CLRB    @(R4)+          ;*TEST INSTRUCTION 400=377 000
 2010  006136  001401                          BEQ     1$              ;BRANCH IF MODE 3 EVEN BYTE CLEARED
 2011                                                                  ; TEST INSTRUCTION FAILED
 2012  006140  104001                          ERROR   +1              ;CPU ERROR
 2013  006142  005304          1$:             DEC     R4              ;REPOSTION POINTER
 2014  006144  005304                          DEC     R4
 2015  006146  105134                          COMB    @(R4)+          ;*TEST INSTRUCTION
 2016  006150  005304                          DEC     R4
 2017  006152  005304                          DEC     R4              ;REPOSITION POINTER
 2018  006154  105234                          INCB    @(R4)+          ;*TEST INSTRUCTION
 2019  006156  001401                          BEQ     3$              ;BRANCH IF GOOD
 2020                                                                  ;MODE 3, EVEN BYTE FAILED
 2021  006160  104001          2$:             ERROR   +1              ;CPU ERROR
 2022  006162  005304          3$:             DEC     R4
 2023  006164  005304                          DEC     R4
 2024  006166  005214                          INC     (R4)            ;R4=401
 2025  006170  105234                          INCB    @(R4)+          ;*TEST INSTRUCTION
 2026  006172  001004                          BNE     4$              ;BRANCH IF 402 NEQ 0
 2027  006174  005304                          DEC     R4
 2028  006176  005304                          DEC     R4              ;R4=401
 2029  006200  105034                          CLRB    @(R4)+          ;401=0
 2030  006202  001401                          BEQ     5$              ;BRANCH IF GOOD
 2031                                                                  ;ODD BYTE FAILED
```

```
2032  006204  104001                  4$:     ERROR   .1              ;CPU ERROR
2033  006206  005304                  5$:     DEC     R4
2034  006210  005304                          DEC     R4              ;R4=401
2035  006212  105134                          COMB    @(R4)+          ;403=377
2036  006214  005304                          DEC     R4
2037  006216  005304                          DEC     R4
2038  006220  105234                          INCB    @(R4)+          ;*TEST INSTRUCTION
2039  006222  001401                          BEQ     7$              ;BRANCH IF GOOD
2040                                                                  ;MODE3 ODD BYTE FAILED.
2041  006224  104001                  6$:     ERROR   .1              ;CPU ERROR
2042  006226                          7$:
2043
2044                                  ;
2045  006226                          MSPL:
2046
2047                                  ;       TEST CLR, COM, DEC    MODE 4
2048  006226  005004                          CLR     R4
2049  006230  105104                          COMB    R4
2050  006232  005204                          INC     R4              ;R4=400
2051  006234  005014                          CLR     (R4)            ;
2052  006236  005124                          COM     (R4)+           ;400=-1
2053  006240  005014                          CLR     (R4)            ;
2054  006242  005224                          INC     (R4)+           ;402=1
2055  006244  005044                          CLR     -(R4)           ;*TEST INSTRUCTION
2056  006246  001401                          BEQ     1$              ;BRANCH IF GOOD
2057                                                                  ;MODE 4 FAILED
2058  006250  104001                          ERROR   .1              ;CPU ERROR
2059  006252  005344                  1$:     DEC     -(R4)           ;*TEST INSTRUCTION 400=-2
2060  006254  005114                          COM     (R4)            ;400=1
2061  006256  001405                          BEQ     2$              ;BRANCH IF BAD
2062  006260  100404                          BMI     2$              ;BRANCH IF BAD
2063  006262  005204                          INC     R4
2064  006264  005204                          INC     R4              ;R4=400
2065  006266  005344                          DEC     -(R4)           ;*TEST INSTRUCTION
2066  006270  001401                          BEQ     3$              ;BRANCH IF GOOD
2067                                                                  ;MODE 4 FAILED
2068  006272  104001                  2$:     ERROR   .1              ;CPU ERROR
2069  006274                          3$:
2070
2071                                  ;
2072  006274                          MSPM:
2073
2074                                  ;       TEST COMB, INCB, CLRB - MODE 4, ODD BYTE
2075  006274  005004                          CLR     R4
2076  006276  105104                          COMB    R4              ;
2077  006300  005204                          INC     R4              ;R4=400
2078  006302  005044                          CLR     -(R4)           ;376=0
2079  006304  105114                          COMB    (R4)
2080  006306  005224                          INC     (R4)+           ;376=001 000
2081  006310  005014                          CLR     (R4)
2082  006312  005124                          COM     (R4)+           ;400=1
2083  006314  005204                          INC     R4              ;R4=403
2084  006316  105044                          CLRB    -(R4)           ; TEST INST. CLEAR ODD BYTE (401)
2085  006320  001401                          BEQ     2$              ;BRANCH IF GOOD
2086                                                                  ;MODE 4 BYTE FAILED
2087  006322  104001                  1$:     ERROR   .1              ;CPU ERROR
```

```
2088  006324  005204                  2$:     INC     R4
2089  006326  005204                          INC     R4              ;R4=403
2090  006330  105144                          COMB    (R4)            ; TEST INST. 401=377
2091  006332  005304                          DEC     R4
2092  006334  005304                          DEC     R4
2093  006336  105244                          INCB    -(R4)           ; TEST INST. 401=0
2094  006340  001401                          BEQ     4$              ;BRANCH IF GOOD
2095                                                                  ;MODE 4 ODD BYTE FAILED
2096  006342  104001                  3$:     ERROR   .1              ;CPU ERROR
2097  006344  105344                  4$:     DECB    -(R4)           ;*TEST INST.
2098  006346  001401                          BEQ     6$              ;BRANCH IF GOOD
2099                                                                  ;MODE 4 DECREMENT ODD BYTE FAILED
2100  006350  104001                  5$:     ERROR   .1              ;CPU ERROR
2101  006352                          6$:
2102
2103                                  ;
2104  006352                          MSPN:
2105
2106                                  ;       TEST CLR, COM, INC - MODE 5
2107  006352  005004                          CLR     R4
2108  006354  005014                          CLR     (R4)
2109  006356  105114                          COMB    (R4)
2110  006360  005224                          INC     (R4).           ;0=400
2111  006362  005054                          CLR     @-(R4)          ;*TEST INST. 400=0
2112  006364  001401                          BEQ     1$              ;BRANCH IF GOOD
2113                                                                  ;MODE 5 FAILED
2114  006366  104001                          ERROR   .1              ;CPU ERROR
2115  006370  005204                  1$:     INC     R4
2116  006372  005204                          INC     R4              ;RESET POINTER TO 0
2117  006374  005154                          COM     @-(R4)          ;*TEST INST. 376=1
2118  006376  001407                          BEQ     2$              ;BRANCH IF BAD
2119  006400  005204                          INC     R4
2120  006402  005204                          INC     R4              ;REPOSITION POINTER
2121  006404  005354                          DEC     @-(R4)          ;*TEST INST. 376=0
2122  006406  001403                          BEQ     2$              ;BRANCH IF BAD
2123  006410  005224                          INC     (R4).           ;0=401 R4=2
2124  006412  105254                          INCB    @-(R4)          ;*TEST INST..400= 0 376
2125  006414  001401                          BEQ     3$              ;BRANCH IF GOOD
2126                                                                  ;MODE 5 FAILED
2127  006416  104001                  2$:     ERROR   .1              ;CPU ERROR
2128  006420                          3$:
2129
2130                                  ;
2131  006420                          MSPO:
2132
2133                                  ;       TEST NEG MODE 5
2134  006420  005004                          CLR     R4
2135  006422  105104                          COMB    R4
2136  006424  005204                          INC     R4              ;R4=400
2137  006426  005001                          CLR     R1
2138  006430  105101                          COMB    R1
2139  006432  005301                          DEC     R1              ;R1=376
2140  006434  005002                          CLR     R2              ;R2=0
2141  006436  005012                          CLR     (R2)            ;0=0
2142  006440  005014                          CLR     (R4)            ;
2143  006442  005114                          COM     (R4)            ;400=-1
```

```
2144  006444  005011                        CLR     (R1)            ;376=0
2145  006446  005454                        NEG     @-(R4)          ;0=0
2146  006450  001401                        BEQ     2$              ;BRANCH IF GOOD
2147                                                                  ;NEG FAILED
2148  006452  104001                  1$:   ERROR   .1              ;CPU ERROR
2149  006454  005334                  2$:   DEC     @(R4)+          ;0=-1
2150  006456  005454                        NEG     @-(R4)          ;0=1
2151  006460  001403                        BEQ     3$              ;BRANCH IF BAD
2152  006462  102402                        BVS     3$              ;BRANCH IF BAD
2153  006464  100401                        BMI     3$              ;BRANCH IF BAD
2154  006466  103401                        BCS     4$              ;BRANCH IF GOOD
2155                                                                  ;NEG FAILED
2156  006470  104001                  3$:   ERROR   .1              ;CPU ERROR
2157  006472  005334                  4$:   DEC     @(R4)+          ; TEST RESULT OF NEGATE
2158  006474  001401                        BEQ     6$              ;BRANCH IF GOOD
2159                                                                  ;RESULT OF NEGATE BAD
2160  006476  104001                  5$:   ERROR   .1              ;CPU ERROR
2161  006500  105212                  6$:   INCB    (R2)            ;0=1
2162  006502  005454                        NEG     @-(R4)          ;0=-1
2163  006504  001403                        BEQ     7$              ;
2164  006506  102402                        BVS     7$
2165  006510  103001                        BCC     7$              ;
2166  006512  100401                        BMI     8$              ;BRANCH IF GOOD
2167                                                                  ;BAD NEGATE
2168  006514  104001                  7$:   ERROR   .1              ;CPU ERROR
2169  006516  105212                  8$:   INCB    (R2)            ;0=0
2170  006520  001401                        BEQ     10$             ;BRANCH IF GOOD
2171  006522  104001                  9$:   ERROR   .1              ;CPU ERROR
2172  006524                        10$:
2173
2174                                  ;
2175
2176  006524                        MSPP:
2177
2178                                  ;       TEST CLR, COM, INC - MODE 6
2179  006524  005004                        CLR     R4
2180  006526  005204                        INC     R4
2181  006530  005204                        INC     R4              ;R4=2
2182  006532  005001                        CLR     R1
2183  006534  105101                        COMB    R1
2184  006536  005201                        INC     R1              ;R1=400
2185  006540  005011                        CLR     (R1)            ;
2186  006542  005121                        COM     (R1)+           ;400=-1
2187  006544  005011                        CLR     (R1)            ;R1=402
2188  006546  005211                        INC     (R1)            ;402=1
2189  006550  005002                        CLR     R2              ;R2=0
2190  006552  005012                        CLR     (R2)            ;0=0
2191  006554  005064  000376                CLR     376(R4)         ;400=0
2192  006560  001401                        BEQ     2$              ;BRANCH IF GOOD
2193  006562  104001                  1$:   ERROR   .1              ;CPU ERROR
2194  006564  005364  000376            2$:   DEC     376(R4)         ;400=-1
2195  006570  005164  000400                COM     400(R4)         ;402=-1
2196  006574  001405                        BEQ     3$              ;BRANCH IF BAD
2197  006576  005264  000400                INC     400(R4)         ;402=-2
2198  006602  005264  000400                INC     400(R4)
2199  006606  001401                        BEQ     4$              ;BRANCH IF GOOD
```

```
                                                                        ;MODE 6 FAILED
006610  104001                  3$:     ERROR   .1              ;CPU ERROR
006612  005261  177776          4$:     INC      2(R1)          ;400=0
006616  001401                          BEQ     6$              ;BRANCH IF GOOD
                                                                        ;INC MODE 6 FAILED
006620  104001                  5$:     ERROR   .1              ;CPU ERROR
006622                          6$:

                                ;
                                ;
                                ;
006622                          MSPQ:

                                ;       TEST NEG MODE 6
006622  005001                          CLR     R1              ;R1=0
006624  005004                          CLR     R4
006626  105104                          COMB    R4
006630  005204                          INC     R4              ;R4=400
006632  005014                          CLR     (R4)            ;
006634  005114                          COM     (R4)            ;400=-1
006636  005044                          CLR     -(R4)           ;376=0
006640  005044                          CLR     -(R4)           ;
006642  005224                          INC     (R4)+           ;374=1 R4=376
006644  005464  000002                  NEG     2(R4)           ;400=1
006650  001403                          BEQ     1$              ;NEGATE FAILED
006652  102402                          BVS     1$
006654  100401                          BMI     1$
006656  103401                          BCS     2$              ;BRANCH IF GOOD
                                                                ;NEGATE FAILED
006660  104001                  1$:     ERROR   .1              ;CPU ERROR
006662  005364  000002          2$:     DEC     2(R4)           ; TEST RESULT OF NEGATE
006666  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;RESULT OF NEGATE FAILED
006670  104001                  3$:     ERROR   .1              ;CPU ERROR
006672  005464  000000          4$:     NEG     0(R4)           ;.0=0
006676  001401                          BEQ     5$              ; BRANCH IF GOOD
                                                                ;NEGATE FAILED
006700  104001                          ERROR   .1              ;CPU ERROR
006702  105461  000374          5$:     NEGB    374(R1)         ;374=0 377
006706  102403                          BVS     6$              ;
006710  001402                          BEQ     6$
006712  100001                          BPL     6$
006714  103401                          BCS     7$              ;BRANCH IF GOOD
                                                                ;NEGATE FAILED
006716  104001                  6$:     ERROR   .1              ;CPU ERROR
006720  105261  000374          7$:     INCB    374(R1)         ;374=0
006724  001401                          BEQ     9$              ;BRANCH IF GOOD
                                                                ;NEGATE FAILED
006726  104001                  8$:     ERROR   .1              ;CPU ERROR
006730                          9$:

                                ;
006730                          MSPR:

                                ;       TEST CLR, COM, INC - MODE 7
006730  005001                          CLR     R1              ;R1=0
```

```
2256  006732  005004                          CLR     R4              ;
2257  006734  105104                          COMB    R4              ;
2258  006736  005204                          INC     R4              ;R4=400
2259  006740  005011                          CLR     (R1)
2260  006742  105111                          COMB    (R1)
2261  006744  005211                          INC     (R1)
2262  006746  005211                          INC     (R1)
2263  006750  005211                          INC     (R1)            ;;0=402
2264  006752  005014                          CLR     (R4)            ;400=0
2265  006754  005064  000002                  CLR     2(R4)           ;
2266  006760  005164  000002                  COM     2(R4)           ;402=-1
2267  006764  005074  177400                  CLR     @-400(R4)       ;402=0
2268  006770  001401                          BEQ     2$              ;BRANCH IF GOOD
2269  006772  104001                  1$:     ERROR   .1              ;CPU ERROR
2270                                                                  ;INSTRUCTION FAILED
2271  006774  005171  000000          2$:     COM     @0(R1)          ;402=-1
2272  007000  100401                          BMI     4$              ;BRANCH IF GOOD
2273  007002  104001                  3$:     ERROR   .1              ;CPU ERROR
2274
2275  007004  005104                  4$:     COM     R4
2276  007006  005274  000401                  INC     @401(R4)                ;402=0
2277  007012  001401                          BEQ     6$              ;BRANCH IF GOOD
2278  007014  104001                  5$:     ERROR   .1              ;CPU ERROR
2279                                                                  ;MODE 7 FAILED
2280  007016                          6$:
2281
2282                                  ;
2283  007016                          MSPS:
2284
2285                                  ;       TEST NEG MODE 7
2286  007016  005004                          CLR     R4
2287  007020  005014                          CLR     (R4)            ;0=0
2288  007022  005002                          CLR     R2              ;
2289  007024  105102                          COMB    R2
2290  007026  005202                          INC     R2              ;R2=400
2291  007030  005012                          CLR     (R2)            ;400=0
2292  007032  005472  177400                  NEG     @-400(R2)               ;NEG OF 0=0
2293  007036  103401                          BCS     1$              ;****
2294  007040  001401                          BEQ     2$              ;BRANCH IF GOOD
2295  007042  104001                  1$:     ERROR   .1              ;CPU ERROR
2296                                                                  ;
2297  007044  005314                  2$:     DEC     (R4)            ;0=-1
2298  007046  005474  000400                  NEG     @.400(R4)               ;0=1
2299  007052  001403                          BEQ     3$              ;BRANCH IF ERROR
2300  007054  102402                          BVS     3$              ;
2301  007056  100401                          BMI     3$              ;
2302  007060  103401                          BCS     4$              ;BRANCH IF GOOD
2303  007062  104001                  3$:     ERROR   .1              ;CPU ERROR
2304                                                                  ;NEGATE MODE 7 FAILED
2305  007064                          4$:
2306
2307                                  ;
2308  007064                          MSPT:
2309
2310                                  ;       TEST SINGLE OPERAND MODE 2 REG 7
2311  007064  005004                          CLR     R4
```

```
2312  007066  105104                    COMB    R4
2313  007070  005204                    INC     R4              ;R4=400
2314  007072  005027                    CLR     (R7)+           ;CLEAR NEXT LOCATION
2315  007074  177777              1$:   .WORD   -1              ;SETUP INITIAL DATA
2316  007076  001401                    BEQ     3$              ;BRANCH IF GOOD
2317  007100  104001              2$:   ERROR   +1              ;CPU ERROR
2318  007102                      3$:
2319
2320                              ;
2321
2322  007102                      MSPU:
2323
2324                              ;       TEST TST MODE 0
2325  007102  005004                    CLR     R4              ;R4=0
2326  007104  000277                    SCC                     ;CONDITION CODES =1111
2327  007106  000244                    CLZ                     ;CC=1011
2328  007110  005704                    TST     R4              ;*TEST INSTRUCTION
2329  007112  103403                    BCS     1$
2330  007114  102402                    BVS     1$
2331  007116  100401                    BMI     1$              ;BRANCH IF ERROR
2332  007120  001401                    BEQ     2$              ;BRANCH IF GOOD
2333  007122  104001              1$:   ERROR   +1              ;CPU ERROR
2334                                                            ;TST MODE 0 FAILED
2335  007124  005304              2$:   DEC     R4              ;R4=-1
2336  007126  000277                    SCC
2337  007130  000250                    CLN                     ;CC=0111
2338  007132  005704                    TST     R4              ;*TEST INSTRUCTION MODE 0
2339  007134  103403                    BCS     3$              ;BRANCH IF ERROR
2340  007136  102402                    BVS     3$
2341  007140  001401                    BEQ     3$
2342  007142  100401                    BMI     4$              ;BRANCH IF GOOD
2343  007144  104001              3$:   ERROR   +1              ;CPU ERROR
2344                                                            ;TST FAILED
2345  007146                      4$:
2346
2347                              ;
2348  007146                      MSPV0:
2349
2350                              ;       TEST TST MODE 0 BYTE
2351  007146  005004                    CLR     R4
2352  007150  105104                    COMB    R4              ;0=000 377
2353  007152  000277                    SCC
2354  007154  000250                    CLN                     ;CC=0111
2355  007156  105704                    TSTB    R4              ;*TEST INSTRUCTION ON EVEN BYTE
2356  007160  102403                    BVS     1$              ;BRANCH IF ERROR
2357  007162  103402                    BCS     1$              ;
2358  007164  102401                    BVS     1$              ;
2359  007166  100401                    BMI     2$              ;BRANCH IF GOOD
2360  007170  104001              1$:   ERROR   +1              ;CPU ERROR
2361                                                            ;
2362  007172  005204              2$:   INC     R4              ;POINT TO 1
2363  007174  105704                    TSTB    R4              ; TEST INSTRUCTION
2364  007176  001401                    BEQ     4$              ;BRANCH IF GOOD
2365  007200  104001              3$:   ERROR   +1              ;CPU ERROR
2366                                                            ;TST FAILED ON BYTE
2367  007202                      4$:
```

```
2368
2369                                    ;
2370  007202                            MSPV:
2371
2372                                    ;       TEST TST MODE 1
2373  007202  005004                            CLR     R4
2374  007204  005014                            CLR     (R4)            ;0=0
2375  007206  000277                            SCC
2376  007210  000244                            CLZ                     ;CC=1011
2377  007212  005714                            TST     (R4)            ;*TEST INSTRUCTION IN MODE 1
2378  007214  103403                            BCS     1$              ;BRANCH IF ERROR
2379  007216  102402                            BVS     1$
2380  007220  100401                            BMI     1$              ;
2381  007222  001401                            BEQ     2$              ;BRANCH IF GOOD
2382  007224  104001                    1$:     ERROR   .1              ;CPU ERROR
2383
2384  007226  005214                    2$.     INC     (R4)            ;0=1
2385  007230  000277                            SCC
2386  007232  005714                            TST     (R4)            ; TEST INSTRUCTION
2387  007234  001403                            BEQ     3$              ;BRANCH IF ERROR
2388  007236  102402                            BVS     3$
2389  007240  103401                            BCS     3$
2390  007242  100001                            BPL     4$              ;BRANCH IF GOOD
2391  007244  104001                    3$:     ERROR   .1              ;CPU ERROR
2392                                                                    ;TST FAILED MODE 1
2393  007246                            4$:
2394
2395                                    ;
2396  007246                            MSPX:
2397
2398                                    ;       TEST TST MODE 1 BYTE
2399  007246  005004                            CLR     R4              ;R4=0
2400  007250  005014                            CLR     (R4)
2401  007252  105114                            COMB    (R4)
2402  007254  005214                            INC     (R4)            ;0=001 000
2403  007256  000277                            SCC
2404  007260  000244                            CLZ                     ;CC=1011
2405  007262  105714                            TSTB    (R4)            ;*TEST INTRUCTION
2406  007264  103403                            BCS     1$              ;BRANCH IF ERROR
2407  007266  102402                            BVS     1$              ;
2408  007270  100401                            BMI     1$
2409  007272  001401                            BEQ     2$              ;BRANCH IF GOOD
2410  007274  104001                    1$:     ERROR   .1              ;CPU ERROR
2411
2412  007276  005204                    2$:     INC     R4              ;R4=1
2413  007300  000277                            SCC
2414  007302  105714                            TSTB    (R4)            ; TEST INSTRUCTION
2415  007304  001403                            BEQ     3$              ;BRANCH IF ERROR
2416  007306  100402                            BMI     3$
2417  007310  102401                            BVS     3$
2418  007312  103001                            BCC     4$              ;BRANCH IF GOOD
2419  007314  104001                    3$:     ERROR   .1              ;CPU ERROR
2420                                                                    ;
2421  007316                            4$:
2422
2423                                    ;
```

```
                                MSPY:

                        ;       TEST TST MODE 2
007316  005004                  CLR     R4              ;
007320  005024                  CLR     (R4)+           ;0=0
007322  005014                  CLR     (R4)
007324  005114                  COM     (R4)            ;2=-1
007326  005004                  CLR     R4              ;R4=0
007330  000277                  SCC             ;
007332  000244                  CLZ             ;CC=1011
007334  005724                  TST     (R4)+           ; TEST INSTRUCTION
007336  103403                  BCS     1$              ;BRANCH IF ERROR
007340  102402                  BVS     1$
007342  100401                  BMI     1$
007344  001401                  BEQ     2$              ;BRANCH IF GOOD
007346  104001          1$:     ERROR   +1              ;CPU ERROR
                                                        ;MODE 2 TEST FAILED
007350  005724          2$:     TST     (R4)+           ;TST LOC2
007352  103403                  BCS     3$
007354  102402                  BVS     3$
007356  001401                  BEQ     3$
007360  100401                  BMI     4$
007362  104001          3$:     ERROR   +1              ;CPU ERROR
                                                        ;MODE 2 FAILED
007364                  4$:

                                ;
007364                  MSPZ:

                        ;       TEST TST MODE 2 BYTE
007364  005004                  CLR     R4
007366  005024                  CLR     (R4)+           ;
007370  105144                  COMB    -(R4)           ;0=377 000
007372  005304                  DEC     R4              ;R4=0
007374  000277                  SCC
007376  000244                  CLZ                     ;CC=1011
007400  105724                  TSTB    (R4)+           ;
007402  102403                  BVS     1$              ;BRANCH IF ERROR
007404  103402                  BCS     1$
007406  100401                  BMI     1$
007410  001401                  BEQ     2$              ;BRANCH IF GOOD
007412  104001          1$:     ERROR   +1              ;CPU ERROR
                                                        ;MODE 2 EVEN BYTE FAILED
007414  000277          2$:     SCC
007416  000250                  CLN                     ;CC=0111
007420  105724                  TSTB    (R4)+           ;
007422  001403                  BEQ     3$              ;
007424  103402                  BCS     3$
007426  102401                  BVS     3$
007430  100401                  BMI     4$              ;BRANCH IF GOOD
007432  104001          3$:     ERROR   +1              ;CPU ERROR
                                                        ;MODE 2 ODD BYTE FAILED
007434                  4$:


007434                  MSPAA:
```

```
2480
2481                                    ;       TEST TST MODE 3
2482  007434  005004                            CLR     R4
2483  007436  005014                            CLR     (R4)
2484  007440  105114                            COMB    (R4)
2485  007442  005214                            INC     (R4)            ;0=400
2486  007444  005034                            CLR     @(R4)+          ;400=0
2487  007446  005004                            CLR     R4              ;R4=0
2488  007450  000277                            SCC
2489  007452  000244                            CLZ                     ;CC=1011
2490  007454  005734                            TST     @(R4)+          ; TEST MODE 3
2491  007456  103403                            BCS     1$              ;BRANCH IF ERROR
2492  007460  102402                            BVS     1$              ;
2493  007462  100401                            BMI     1$
2494  007464  001401                            BEQ     2$              ;BRANCH IF GOOD
2495  007466  104001                    1$:     ERROR   +1              ;CPU ERROR
2496                                                                    ;MODE 3 FAILED
2497  007470  005304                    2$:     DEC     R4
2498  007472  005304                            DEC     R4              ;R4=0
2499  007474  005334                            DEC     @(R4)+          ;400=-1
2500  007476  005004                            CLR     R4
2501  007500  000277                            SCC
2502  007502  000250                            CLN                     ;CC=0111
2503  007504  005734                            TST     @(R4)+          ; TEST INSTRUCTION
2504  007506  103403                            BCS     3$              ;
2505  007510  001402                            BEQ     3$              ;BRANCH IF ERROR
2506  007512  102401                            BVS     3$              ;
2507  007514  100401                            BMI     4$              ;BRANCH IF GOOD
2508                                                                    ;ERROR MODE 3
2509  007516  104001                    3$:     ERROR   +1              ;CPU ERROR
2510  007520                            4$:
2511
2512                                    ;
2513  007520                            MSP88:
2514
2515                                    ;       TEST TST MODE 3 AUTO-INC
2516  007520  005004                            CLR     R4
2517  007522  005014                            CLR     (R4)            ;0=0
2518  007524  105114                            COMB    (R4)            ;
2519  007526  005214                            INC     (R4)            ;0=400
2520  007530  005001                            CLR     R1
2521  007532  105101                            COMB    R1
2522  007534  005201                            INC     R1              ;R1=400
2523  007536  005011                            CLR     (R1)            ;400=0
2524  007540  000277                            SCC
2525  007542  005734                            TST     @(R4)+          ;400=0
2526  007544  103403                            BCS     1$              ;ERROR IF CARRY
2527  007546  102402                            BVS     1$              ;ERROR IF OVERFLOW
2528  007550  100401                            BMI     1$              ;ERROR IF MINUS
2529  007552  001401                            BEQ     2$              ;ERROR IF NOT EQUAL
2530  007554  104001                    1$:     ERROR   +1              ;CPU ERROR
2531                                                        ;CC SHOULD = 0100
2532  007556  005304                    2$:     DEC     R4
2533  007560  005304                            DEC     R4
2534  007562  005704                            TST     R4              ;SEE IF AUTO-INC WORKED
2535  007564  001401                            BEQ     4$              ;ERROR IF R4 NE 0
```

```
2536  007566  104001                  3$:     ERROR   .1              ;CPU ERROR
2537                                                          ;AUTO-INC FIALED
2538  007570                          4$:
2539
2540                                  ;
2541  007570                          MSTB3:
2542
2543                                  ;       TEST TST MODE 3 BYTE
2544  007570  005004                          CLR     R4
2545  007572  005014                          CLR     (R4)
2546  007574  105114                          COMB    (R4)
2547  007576  005214                          INC     (R4)
2548  007600  005214                          INC     (R4)            ;0=401
2549  007602  005001                          CLR     R1
2550  007604  105101                          COMB    R1
2551  007606  005201                          INC     R1              ;R1=400
2552  007610  005011                          CLR     (R1)
2553  007612  005111                          COM     (R1)
2554  007614  105011                          CLRB    (R1)            ;400=377 000
2555  007616  105734                          TSTB    @(R4)+          ;** TEST INSTRUCTION
2556  007620  001403                          BEQ     1$              ;ERROR IF EQUAL
2557  007622  103402                          BCS     1$              ;ERROR IF CARRY SET
2558  007624  102401                          BVS     1$              ;ERROR IR OVERFLOW
2559  007626  100401                          BMI     2$              ;BRANCH IF MINUS
2560  007630  104001                  1$:     ERROR   .1              ;CPU ERROR
2561                                                          ;CC ERROR
2562  007632  005304                  2$:     DEC     R4
2563  007634  005304                          DEC     R4
2564  00763,  001401                          BEQ     4$                      ;BRANCH IF AUTO-INC WORKED
2565  007640  104001                  3$:     ERROR   .1              ;CPU ERROR
2566                                                          ;AUTO-INC FAILED
2567  007642                          4$:
2568
2569                                  ;
2570  007642                          MST4:
2571
2572                                  ;       TEST TST MODE 4
2573  007642  005004                          CLR     R4
2574  007644  005014                          CLR     (R4)            ;0=0
2575  007646  005204                          INC     R4
2576  007650  005204                          INC     R4              ;R4=2
2577  007652  000277                          SCC
2578  007654  000244                          CLZ                     ;CC=1011
2579  007656  005744                          TST     -(R4)           ;**TEST INTRUCTION
2580  007660  103403                          BCS     1$              ;ERROR IF CARRY
2581  007662  102402                          BVS     1$              ;ERROR IF OVERFLOW
2582  007664  100401                          BMI     1$              ;ERROR IF MINUS
2583  007666  001401                          BEQ     2$              ;BRANCH IF GOOD
2584  007670  104001                  1$:     ERROR   .1              ;CPU ERROR
2585                                                          ;CC WRONG
2586  007672  005704                  2$:     TST     R4              ;INSURE CORRECT AUTO-DEC
2587  007674  001401                          BEQ     4$              ;BRANCH IF GOOD AUTO-DEC
2588                                                                  ;BAD AUTO-DEC
2589  007676  104001                  3$:     ERROR   .1              ;CPU ERROR
2590  007700                          4$:
2591
```

```
2592                                    ;
2593  007700                            MST4B:
2594
2595                                    ;       TEST TST MODE 4 BYTE
2596  007700  005004                           CLR     R4
2597  007702  005014                           CLR     (R4)
2598  007704  005114                           COM     (R4)
2599  007706  105114                           COMB    (R4)            ;0=377 000
2600  007710  000277                           SCC
2601  007712  005204                           INC     R4
2602  007714  005204                           INC     R4              ;R4=2
2603  007716  105744                           TSTB    -(R4)           ;**TEST INSTRUCTION
2604  007720  001403                           BEQ     1$              ;ERROR IF EQUAL TO 0
2605  007722  103402                           BCS     1$              ;ERROR IF CARRY
2606  007724  102401                           BVS     1$              ;ERROR IF OVERFLOW
2607  007726  100401                           BMI     2$              ;BRANCH IF MINUS
2608  007730  104001                   1$:     ERROR   +1              ;CPU ERROR
2609                                                          ;CC SHOULD EQUAL 0100
2610  007732  105744                   2$:     TSTB    -(R4)           ;**TEST EVEN BYTE
2611  007734  001401                           BEQ     4$              ;BRANCH IF GOOD
2612  007736  104001                   3$:     ERROR   +1              ;CPU ERROR
2613                                                          ;CC SHOULD EQUAL 0100 AND R4=-1
2614  007740                            4$:
2615                                    ;
2616  007740                            MST5:
2617
2618                                    ;       TEST TST MODE 5
2619  007740  005004                           CLR     R4              ;
2620  007742  005024                           CLR     (R4)+           ;0=0, R4=2
2621  007744  000277                           SCC
2622  007746  000244                           CLZ                     ;CC=1011
2623  007750  005754                           TST     @-(R4)          ; TEST INSTRUCTION
2624  007752  103403                           BCS     1$              ;ERROR IF CARRY
2625  007754  102402                           BVS     1$              ;ERROR IF OVERFLOW
2626  007756  100401                           BMI     1$              ;ERROR IF MINUS
2627  007760  001401                           BEQ     2$              ;BRANCH IF GOOD
2628  007762  104001                   1$:     ERROR   +1              ;CPU ERROR
2629                                                          ;CC WRONG, SHOULD = 0100
2630  007764  005704                   2$:     TST     R4
2631  007766  001401                           BEQ     4$              ;BRANCH IF AUTO-DEC WORKED
2632  007770  104001                   3$:     ERROR   +1              ;CPU ERROR
2633                                                          ;AUTO-DEC FAILED
2634  007772                            4$:
2635
2636                                    ;
2637  007772                            MST5B:
2638
2639                                    ;       TEST TST MODE 5 BYTE
2640  007772  005004                           CLR     R4
2641  007774  005014                           CLR     (R4)
2642  007776  105114                           COMB    (R4)
2643  010000  005214                           INC     (R4)            ;0=400
2644  010002  005034                           CLR     @(R4)+          ;400=0, R4=2
2645  010004  005154                           COM     @-(R4)          ;
2646  010006  105134                           COMB    @(R4)+          ;400=377 000 R4=2
2647  010010  105754                           TSTB    @-(R4)          ;**TEST INSTRUCTION
```

```
2648  010012  103403                    BCS     1$              ;ERROR IF CARRY
2649  010014  100402                    BMI     1$              ;ERROR IF MINUS
2650  010016  102401                    BVS     1$              ;ERROR IF OVERFLOW
2651  010020  001401                    BEQ     2$              ;BRANCH IF GOOD
2652  010022  104001            1$:     ERROR   .1              ;CPU ERROR
2653                                                    ;CC SHOULD = 0100
2654  010024  005224            2$:     INC     (R4)+           ;0=401
2655  010026  105754                    TSTB    @-(R4)          ;**TEST INSTRUCTION
2656  010030  100401                    BMI     4$              ;BRANCH IF GOOD
2657                                                            ;EVEN BYTE FAILURE
2658  010032  104001            3$:     ERROR   .1              ;CPU ERROR
2659  010034                    4$:
2660
2661                              ;
2662  010034                    MST6:
2663
2664                              ;       TEST TST MODE 6
2665  010034  005004                    CLR     R4
2666  010036  005014                    CLR     (R4)            ;0=0
2667  010040  105104                    COMB    R4
2668  010042  005204                    INC     R4              ;R4=400
2669  010044  005014                    CLR     (R4)
2670  010046  005114                    COM     (R4)            ;400=-1
2671  010050  005764  177400           TST     -400(R4)        ;**TEST LOCATION 0
2672  010054  103403                    BCS     1$              ;ERROR IF CARRY
2673  010056  102402                    BVS     1$              ;ERROR IF OVERFLOW
2674  010060  100401                    BMI     1$              ;ERROR IF MINUS
2675  010062  001401                    BEQ     2$              ;BRANCH IF ZERO
2676  010064  104001            1$:     ERROR   .1              ;CPU ERROR
2677                                                    ;CC ARE WRONG
2678  010066  005004            2$:     CLR     R4              ;
2679  010070  005764  000400           TST     400(R4)         ;TST LOCATION 400
2680  010074  001401                    BEQ     3$              ;ERROR IF EQUAL
2681  010076  100401                    BMI     4$              ;BRANCH IF MINUS
2682  010100  104001            3$:     ERROR   .1              ;CPU ERROR
2683                                                    ;CC ERROR
2684  010102                    4$:
2685
2686
2687  010102                    MST7:
2688
2689                              ;       TEST TST MODE 7
2690  010102  005004                    CLR     R4
2691  010104  005014                    CLR     (R4)
2692  010106  005124                    COM     (R4)+           ;0=-1
2693  010110  005014                    CLR     (R4)            ;2=0
2694  010112  005002                    CLR     R2              ;R2=0
2695  010114  005004                    CLR     R4
2696  010116  105104                    COMB    R4
2697  010120  005204                    INC     R4              ;R4=400
2698  010122  005014                    CLR     (R4)            ;400=0
2699  010124  005774  177402           TST     @-376(R4)       ;**TEST LOCATION 0
2700  010130  103403                    BCS     1$              ;ERROR IF CARRY
2701  010132  102402                    BVS     1$              ;ERROR IF OVERFLOW
2702  010134  001401                    BEQ     1$              ;ERROR IF ZERO
2703  010136  100401                    BMI     2$              ;BRANCH IF GOOD
```

```
2704  010140  104001                  1$:     ERROR   +1              ;CPU ERROR
2705                                                          ;ERROR IN CC
2706  010142  005222                  2$:     INC     (R2)+           ;0=-2 R2=2
2707  010144  005774  177402                  TST     @-376(R4)       ;** CHECK CONTENTS OF LOCATION 2
2708  010150  100401                          BMI     3$              ;ERROR IF MINUS
2709  010152  001401                          BEQ     4$              ;BRANCH IF GOOD
2710  010154  104001                  3$:     ERROR   +1              ;CPU ERROR
2711                                                          ;CC SHOULD = 0100
2712  010156                          4$:
2713
2714                                  ;
2715                                  ;
2716                                  ;
2717                                  ;
2718                                  ;
2719                                  .SBTTL ************ DOUBLE OPERAND TESTS ******************
2720                                  ;
2721  010156                          MDMO:
2722
2723                                  ;       TEST MOVE MODE 0
2724  010156  005004                          CLR     R4              ;R4=0
2725  010160  005001                          CLR     R1
2726  010162  005101                          COM     R1              ;R1=-1
2727  010164  010104                          MOV     R1,R4           ;**TEST MOVE INSTRUCTION
2728  010166  001402                          BEQ     1$              ;ERROR IF R4=0
2729  010170  102401                          BVS     1$              ;ERROR IF OBERFLOW
2730  010172  100401                          BMI     2$              ;BRANCH IF MINUS
2731  010174  104001                  1$:     ERROR   +1              ;CPU ERROR
2732                                                          ;CC SHOULD = 100-, R4 SHOULD=-1
2733  010176  005204                  2$:     INC     R4              ;R4 SHOULD =0
2734  010200  001375                          BNE     1$              ;ERROR IF R4 NE 0
2735
2736                                  ;
2737  010202                          MDAO:
2738
2739                                  ;       TEST ADD MODE 0
2740  010202  005004                          CLR     R4              ;R4=0
2741  010204  005001                          CLR     R1
2742  010206  005101                          COM     R1              ;R1=-1
2743  010210  060104                          ADD     R1,R4           ;** TEST ADD OF R1 TO R4
2744  010212  001403                          BEQ     1$              ;ERROR IF ZERO
2745  010214  103402                          BCS     1$              ;ERROR IF CARRY
2746  010216  102401                          BVS     1$              ;ERROR IF OVERFLOW
2747  010220  100401                          BMI     2$              ;BRANCH IF MINUS
2748  010222  104001                  1$:     ERROR   +1              ;CPU ERROR
2749                                                          ;CC SHOULD = 1000 , R4=-1
2750  010224  005204                  2$:     INC     R4              ;R4 SHOULD =0
2751  010226  001401                          BEQ     4$              ;BRANCH IF R4=0
2752  010230  104001                  3$:     ERROR   +1              ;CPU ERROR
2753                                                          ;R4 SHOULD = 0
2754  010232                          4$:
2755
2756                                  ;
2757  010232                          MDSO:
2758
2759                                  ;       TEST SUB MODE 0
```

```
 2760  010232  005004                          CLR     R4
 2761  010234  005001                          CLR     R1
 2762  010236  005201                          INC     R1                      ;R1=1 R4=0
 2763  010240  160104                          SUB     R1,R4                   ;**TEST OF R4-R1, R4= 1
 2764  010242  102403                          BVS     1$                      ;ERROR IF V SET
 2765  010244  103002                          BCC     1$                      ;ERROR IF NO CARRY
 2766  010246  001401                          BEQ     1$                      ;ERROR IF =0
 2767  010250  100401                          BMI     2$                      ;BRANCH IF MINUS
 2768  010252  104001                  1$:     ERROR   ;1              ;CPU ERROR
 2769                                          ;CC SHOULD = 1001
 2770  010254  005101                  2$:     COM     R1                      ;R1=1
 2771  010256  005201                          INC     R1                      ;GET TWO'S COMPLIMENT, R1=-1
 2772  010260  160104                          SUB     R1,R4                   ;**TEST R4-R1 (1- 1= 0
 2773  010262  001401                          BEQ     4$                      ;BRANCH IF ZERO
 2774  010264  104001                  3$:     ERROR   ;1              ;CPU ERROR
 2775                                          ;CC SHOULD = 0100
 2776  010266                          4$:
 2777
 2778                                  ;
 2779  010266                          MOM27:
 2780
 2781                                  ;       TEST MOV MODE 27,00
 2782  010266  000257                          CCC                             ;CC=0000
 2783  010270  012704  125252                  MOV     #125252,R4              ;**TEST MOVE
 2784  010274  001401                          BEQ     1$                      ;ERROR IF = 0
 2785  010276  100401                          BMI     2$                      ;BRANCH IF MINUS
 2786  010300  104001                  1$:     ERROR   ;1              ;CPU ERROR
 2787                                          ;CC SHOULD = 1000
 2788  010302  012701  052525          2$:     MOV     #052525,R1              ;**TEST MOVE
 2789  010306  100401                          BMI     3$                      ;ERROR IF MINUS
 2790  010310  001001                          BNE     4$                      ;BRANCH IF NE 0
 2791  010312  104001                  3$:     ERROR   ;1              ;CPU ERROR
 2792                                          ;CC SHOULD = 0000
 2793  010314  060104                  4$:     ADD     R1,R4                   ;R1+R4=-1
 2794  010316  100401                          BMI     6$                      ;BRANCH IF MINUS
 2795  010320  104001                  5$:     ERROR   ;1              ;CPU ERROR
 2796                                          ;MOV FAILED
 2797  010322  005204                  6$:     INC     R4                      ;R4+1=0
 2798  010324  001375                          BNE     5$                      ;ERROR IF NOT ZERO
 2799
 2800                                  ;
 2801  010326                          MBI00:
 2802
 2803                                  ;       TEST BIC, BIS MODE 0,0
 2804  010326  005004                          CLR     R4
 2805  010330  005104                          COM     R4                      ;R4=-1
 2806  010332  012701  125252                  MOV     #125252,R1              ;SETUP R1 TEST DATA
 2807  010336  012702  052525                  MOV     #052525,R2              ;R2=COMPLIMENT OF R1
 2808  010342  000261                          SEC
 2809  010344  040104                          BIC     R1,R4                   ;**TEST BIC WITH CARRY SET
 2810  010346  103003                          BCC     1$                      ;ERROR IF NO CARRY
 2811  010350  102402                          BVS     1$                      ;ERROR IF OVERFLOW
 2812  010352  001401                          BEQ     1$                      ;ERROR IF 0
 2813  010354  100001                          BPL     2$                      ;BRANCH IF PLUS
 2814  010356  104001                  1$:     ERROR   ;1              ;CPU ERROR
 2815                                          ;CC SHOULD = 0001
```

```
2816  010360  020402                  2$:     CMP     R4,R2                   ;COMPARE CONTENTS OF R4 AND R2
2817  010362  001401                          BEQ     4$                      ;BRANCH IF EQUAL
2818  010364  104001                  3$:     ERROR   .1              ;CPU ERROR
2819                                                                  ;R4 AND R2 SHOULD BE EQUAL
2820  010366  005301                  4$:     DEC     R1                      ;R1=125251
2821  010370  050201                          BIS     R2,R1                   ;BIS 052525 AND 125251=177775
2822  010372  100401                          BMI     6$                      ;BRANCH IF MIUS VALUE
2823  010374  104001                  5$:     ERROR   .1              ;CPU ERROR
2824                                                                  ;BAD BIS OPERATION
2825  010376  005201                  6$:     INC     R1
2826  010400  005201                          INC     R1
2827  010402  005201                          INC     R1                      ;R1=0
2828  010404  001373                          BNE     5$                      ;ERROR IF NE 0
2829
2830                                  ;
2831  010406                          MBC00:
2832
2833                                  ;       TEST BIT, CMP MODE 0,0
2834  010406  012701  125252                  MOV     #125252,R1              ;R1=125252
2835  010412  012704  100000                  MOV     #100000,R4              ;R4=100000
2836  010416  012702  052525                  MOV     #052525,R2              ;R2=052525
2837  010422  030401                          BIT     R4,R1                   ;**TEST OF BIT .CC=1000
2838  010424  001401                          BEQ     1$                      ;ERROR IF EQ 0
2839  010426  100401                          BMI     2$                      ;BRANCH IF GOOD
2840  010430  104001                  1$:     ERROR   .1              ;CPU ERROR
2841                                                                  ;CC SHOULD = 1000
2842  010432  020401                  2$:     CMP     R4,R1                   ;*TEST 100000-125252=25252
2843  010434  001402                          BEQ     3$                      ;ERROR IF EQUAL 0
2844  010436  103001                          BCC     3$                      ;ERROR IF CARRY CLEARED
2845  010440  100401                          BMI     4$                      ;BRANCH IF GOOD
2846  010442  104001                  3$:     ERROR   .1              ;CPU ERROR
2847                                                                  ;CC SHOULD = 0010
2848  010444  020104                  4$:     CMP     R1,R4                   ;125252-100000 = 25252
2849  010446  001403                          BEQ     5$                      ;ERROR IF EQUAL
2850  010450  103402                          BCS     5$                      ;ERROR IF CARRY
2851  010452  102401                          BVS     5$                      ;ERROR IF OVERFLOW
2852  010454  100001                          BPL     6$                      ;BRANCH IF GOOD
2853  010456  104001                  5$:     ERROR   .1              ;CPU ERROR
2854                                                                  ;CC SHOULD =0001
2855  010460  005004                  6$:     CLR     R4
2856  010462  005204                          INC     R4                      ;R4=1
2857  010464  000277                          SCC
2858  010466  030401                          BIT     R4,R1                   ;R4 ? R1 = 2
2859  010470  001401                          BEQ     8$                      ;BRANCH IF GOOD
2860  010472  104001                  7$:     ERROR   .1              ;CPU ERROR
2861                                                                  ;CC SHOULD = 0101
2862  010474                          8$:
2863
2864                                  ;
2865  010474                          MM11:
2866
2867                                  ;       TEST MOV, MOVB MODE 1,1 AND SIGN EXT ON MOVB TO GPR
2868  010474  012704  000400                  MOV     #400,R4                 ;R4=400
2869  010500  012701  000402                  MOV     #402,R1                 ;R1=402
2870  010504  005014                          CLR     (R4)                    ;
2871  010506  005114                          COM     (R4)                    ;400=-1
```

```
2872  010510  005011                          CLR     (R1)                    ;
2873  010512  105111                          COMB    (R1)            ;402=000'377
2874  010514  005002                          CLR     R2              ;R2=0
2875  010516  012703  000405                 MOV     #405,R3         ;R3=405
2876  010522  000277                          SCC                     ;CC=1111
2877  010524  011412                          MOV     (R4),(R2)       ;MOV    400 TO 0 ,0=-1
2878  010526  001403                          BEQ     1$              ;ERROR IF 0
2879  010530  102402                          BVS     1$              ;ERROR IF OVERFLOW
2880  010532  103001                          BCC     1$              ;ERROR IF NO CARRY
2881  010534  100401                          BMI     2$              ;BRANCH IF GOOD
2882  010536  104001                  1$:     ERROR   #1              ;CPU ERROR
2883                                                          ;CC SHOULD =1001
2884  010540  005212                  2$:     INC     (R2)            ;0=0
2885  010542  001004                          BNE     3$              ;ERROR IF NOT 0
2886  010544  000257                          CCC                     ;CC=0000
2887  010546  111113                          MOVB    (R1),(R3)               ;405=377
2888  010550  001401                          BEQ     3$              ;ERROR IF EQUAL
2889  010552  100401                          BMI     4$              ;BRANCH IF GOOD
2890  010554  104001                  3$:     ERROR   #1              ;CPU ERROR
2891                                                          ;
2892  010556  105213                  4$:     INCB    (R3)            ;405=0
2893  010560  001375                          BNE     3$              ;ERROR IF 405 NOT 0
2894                                  ;CHECK THAT SIGN EXTENSION OCCURS ON A MOVB TO GENERAL REGISTER.
2895  010562  005002                          CLR     R2              ;INIT R2 TO ZERO.
2896  010564  111102                          MOVB    (R1),R2         ;MOVE 377 TO R2
2897  010566  100005                          BPL     5$              ;ERROR! BIT 15 SHOULD BE SET.
2898  010570  102404                          BVS     5$              ;V BIT SHOULD BE CLEARED
2899  010572  103403                          BCS     5$              ;CARRY BIT SHOULD BE UNAFFECTED
2900  010574  022702  177777                 CMP     #177777,R2      ;TEST R2
2901  010600  001401                          BEQ     6$              ;SIGN EXTENDED THROUGH UPPER BYTE
2902                                                                  ;ERROR! BYTE SHOULD HAVE
2903                                                                  ;SIGN EXTENDED THROUGH UPPER BYTE
2904  010602  104001                  5$:     ERROR   #1              ;CPU ERROR
2905  010604                          6$:
2906
2907                                  ;
2908  010604                          MA11:
2909
2910                                  ;       TEST ADD MODE 1.1
2911  010604  012704  000400                  MOV     #400,R4         ;R4=400
2912  010610  012701  000402                  MOV     #402,R1         ;R1=402
2913  010614  012714  177753                  MOV     #-25,(R4)       ;400=-25
2914  010620  012711  000024                  MOV     #24,(R1)        ;402=24
2915  010624  061114                          ADD     (R1),(R4)       ;-25+24=-1
2916  010626  001404                          BEQ     1$              ;ERROR IF 0
2917  010630  103403                          BCS     1$              ;ERROR IF CARRY
2918  010632  100002                          BPL     1$              ;ERROR IF POSITIVE RESULT
2919  010634  005214                          INC     (R4)            ;-1+1=0
2920  010636  001401                          BEQ     2$              ;BRANCH IF GOOD
2921  010640  104001                  1$:     ERROR   #1              ;CPU ERROR
2922                                                          ;CC SHOULD = 1000
2923  010642                          2$:
2924
2925                                  ;
2926  010642                          MS11:
2927
```

```
2928                                    ;       TEST SUB MODE 1,1
2929  010642  012704  000400                    MOV     #400,R4         ;R4=400
2930  010646  012701  000404                    MOV     #404,R1         ;R1=404
2931  010652  012714  000003                    MOV     #3,(R4)         ;400=3
2932  010656  012711  000006                    MOV     #6,(R1)         ;406=6
2933  010662  000277                            SCC                     ;CC=1111
2934  010664  161411                            SUB     (R4),(R1)       ;6-3=3
2935  010666  001402                            BEQ     1$              ;ERROR IF 0
2936  010670  100401                            BMI     1$              ;ERROR IF MINUS
2937  010672  103001                            BCC     2$              ;BRANCH IF GOOD
2938  010674  104001                    1$:     ERROR   +1              ;CPU ERROR
2939                                                            ;CC SHOULD = 0000
2940  010676  161411                    2$:     SUB     (R4),(R1)       ;3-3=0
2941  010700  001375                            BNE     1$              ;ERROR IF NOT 0
2942
2943                                    ;
2944  010702                            MB811:
2945
2946                                    ;       TEST BIC, BIS MODE 1,1
2947  010702  012704  000400                    MOV     #400,R4         ;R4=400
2948  010706  012701  000402                    MOV     #402,R1         ;R1=402
2949  010712  012714  052525                    MOV     #052525,(R4)    ;400=052525
2950  010716  012711  125252                    MOV     #125252,(R1)    ;402=125252
2951  010722  051411                            BIS     (R4),(R1)       ;R4 V R1 = -1
2952  010724  001401                            BEQ     1$              ;ERROR IF 0
2953  010726  100401                            BMI     2$              ;BRANCH IF GOOD
2954  010730  104001                    1$:     ERROR   +1              ;CPU ERROR
2955                                                            ;CC SHOULD = 1000
2956  010732  005211                    2$:     INC     (R1)            ;402=0
2957  010734  001401                            BEQ     4$              ;BRANCH IF GOOD
2958  010736  104001                    3$:     ERROR   +1              ;CPU ERROR
2959                                                            ;CC SHOULD = 0100
2960  010740  005311                    4$:     DEC     (R1)            ;402=-1
2961  010742  041411                            BIC     (R4),(R1)       ;R1=125252
2962  010744  001401                            BEQ     5$              ;ERROR IF 0
2963  010746  100401                            BMI     6$              ;BRANCH IF GOOD
2964  010750  104001                    5$:     ERROR   +1              ;CPU ERROR
2965                                                            ;CC SHOULD = 1000
2966  010752  005111                    6$:     COM     (R1)            ;402=052525
2967  010754  041114                            BIC     (R1),(R4)       ;400=0
2968  010756  001401                            BEQ     8$              ;BRANCH IF GOOD
2969  010760  104001                    7$:     ERROR   +1              ;CPU ERROR
2970                                                            ;CC SHOULD = 0100
2971  010762                            8$:
2972
2973                                    ;
2974  010762                            MBC11:
2975
2976                                    ;       TEST BIT, CMP MODE 1,1
2977  010762  012704  000400                    MOV     #400,R4         ;R4=400
2978  010766  012714  052525                    MOV     #052525,(R4)    ;400=052525
2979  010772  012701  000402                    MOV     #402,R1         ;R1=402
2980  010776  012711  125252                    MOV     #125252,(R1)    ;402=125252
2981  011002  000241                            CLC                     ;CLEAR CARRY
2982  011004  031411                            BIT     (R4),(R1)       ;*052525^125252=0
2983  011006  103401                            BCS     1$              ;ERROR IF CARRY
```

```
 2984  011010  001401                        BEQ     2$                      ;BRANCH IF GOOD
 2985  011012  104001              1$:       ERROR   .1              ;CPU ERROR
 2986                                        ;CC SHOULD = 0100
 2987  011014  021411              2$:       CMP     (R4),(R1)               ;400-402=125253
 2988  011016  001403                        BEQ     3$                      ;ERROR IF ZERO
 2989  011020  103002                        BCC     3$                      ;ERROR IF NO CARRY
 2990  011022  102001                        BVC     3$                      ;ERROR IF NO OVERFLOW
 2991  011024  100401                        BMI     4$                      ;BRANCH IF GOOD
 2992  011026  104001              3$:       ERROR   .1              ;CPU ERROR
 2993                                        ;CC SHOULD = 1000
 2994  011030  005014              4$:       CLR     (R4)
 2995  011032  005214                        INC     (R4)                    ;400=1
 2996  011034  031114                        BIT     (R1),(R4)               ;125252!1=0
 2997  011036  001401                        BEQ     6$                      ;BRANCH IF GOOD
 2998  011040  104001              5$:       ERROR   .1              ;CPU ERROR
 2999                                        ;CC SHOULD= 0100
 3000  011042                    6$:
 3001
 3002                              ;
 3003  011042                    MM22:
 3004
 3005                              ;       TEST MOV MODE 2,2
 3006  011042  012704  000400                MOV     #400,R4         ;R4=400
 3007  011046  012701  000402                MOV     #402,R1         ;R1=402
 3008  011052  012714  000005                MOV     #5,(R4)         ;400=5
 3009  011056  005021                        CLR     (R1)+           ;402=0
 3010  011060  005011                        CLR     (R1)            ;
 3011  011062  005111                        COM     (R1)            ;404=-1
 3012  011064  005741                        TST     -(R1)           ;R1=402
 3013  011066  000277                        SCC                     ;CC=1111
 3014  011070  012124                        MOV     (R1)+,(R4)+     ;400=0 R4=402 R1=404
 3015  011072  100403                        BMI     1$              ;ERROR IF MINUS
 3016  011074  103002                        BCC     1$              ;ERROR IF NO CARRY
 3017  011076  102401                        BVS     1$              ;ERROR IF OVERFLOW
 3018  011100  001401                        BEQ     2$              ;BRANCH IF GOOD
 3019  011102  104001              1$:       ERROR   .1              ;CPU ERROR
 3020                                                ;CC SHOULD= 0101
 3021  011104  005244              2$:       INC     -(R4)           ;400=1 R4=400
 3022
 3023  011106  061411                        ADD     (R4),(R1)       ;1+ -1 =0
 3024  011110  001401                        BEQ     4$              ;BRANCH IF GOOD
 3025  011112  104001              3$:       ERROR   .1              ;CPU ERROR
 3026                                                ;CC SHOULD = 0100
 3027  011114                    4$:
 3028
 3029                              ;
 3030  011114                    MS22:
 3031
 3032                              ;       TEST SUB MODE 2,2
 3033  011114  012704  000400                MOV     #400,R4                 ;R4=400
 3034  011120  012701  000402                MOV     #402,R1                 ;R1=402
 3035  011124  012714  177760                MOV     #177760,(R4)            ;400=177760
 3036  011130  012711  177750                MOV     #177750,(R1)            ;402=177750
 3037  011134  162421                        SUB     (R4)+,(R1)+             ;R1=177770
 3038  011136  001403                        BEQ     1$                      ;ERROR IF ZERO
 3039  011140  102402                        BVS     1$                      ;ERROR IF OVERFLOW
```

```
                        ;************ DOUBLE OPERAND TESTS *****************                SEQ 0058

3040  011142  103001                    BCC     1$                      ;ERROR IF NO CARRY
3041  011144  100401                    BMI     2$                      ;BRANCH IF GOOD
3042  011146  104001              1$:   ERROR   ^1              ;CPU ERROR
3043                                                          ;CC SHOULD=1000
3044  011150  005241              2$:   INC     -(R1)                   ;R1=1777771
3045  011152  162721  177771            SUB     #177771,(R1)+           ;R1=0
3046  011156  100401                    BMI     3$                      ;ERROR IF MINUS
3047  011160  001401                    BEQ     4$                      ;BRANCH IF GOOD
3048  011162  104001              3$:   ERROR   ^1              ;CPU ERROR
3049                                                          ;CCSHOULD = 0100
3050  011164                       4$:
3051
3052                               ;
3053  011164                       MB822:
3054
3055                               ;       TEST BIC, BICB, BIS, BISB MODE 2,2
3056  011164  012704  000400              MOV     #400,R4         ;R4=400
3057  011170  012701  000402              MOV     #402,R1         ;R1=402
3058  011174  012702  000404              MOV     #404,R2         ;R2=404
3059  011200  012714  141401              MOV     #141401,(R4)    ;400=303 001
3060  011204  012711  177405              MOV     #177405,(R1)    ;402=377 005
3061  011210  012722  000070              MOV     #70,(R2)+       ;404=2070
3062  011214  012722  177777              MOV     #-1,(R2)+       ;406=-1
3063  011220  042421                      BIC     (R4)+,(R1)+     ;402=074004
3064  011222  001401                      BEQ     1$              ;ERROR IF ZERO
3065  011224  100001                      BPL     2$              ;BRANCH IF GOOD
3066                                                          ;CC SHOULD = 1000
3067  011226  104001              1$:   ERROR   ^1              ;CPU ERROR
3068  011230  052421              2$:   BIS     (R4)+,(R1)+     ;404=074074
3069  011232  142421                      BICB    (R4)+,(R1)+     ;406=074
3070  011234  005301                      DEC     R1              ;R4=405 R1=406
3071  011236  152421                      BISB    (R4)+,(R1)+     ;406=-1 R4=406 R1=407
3072  011240  100401                      BMI     4$              ;BRANCH IF GOOD
3073                                                          ;406 SHOULD=-1
3074  011242  104001              3$:   ERROR   ^1              ;CPU ERROR
3075  011244  005214              4$:   INC     (R4)            ;406 SHOULD=0
3076  011246  001401                      BEQ     6$              ;BRANCH IF GOOD
3077                                                          ;ERROR! 406 NE 0
3078  011250  104001              5$:   ERROR   ^1              ;CPU ERROR
3079  011252                       6$:
3080
3081                               ;
3082  011252                       MBC22:
3083
3084                               ;       TEST BIT, CMP MODE 2,2
3085  011252  012704  000400              MOV     #400,R4         ;R4=400
3086  011256  012701  000402              MOV     #402,R1         ;R1=402
3087  011262  012714  125252              MOV     #125252,(R4)    ;400=125252
3088  011266  012721  100001              MOV     #100001,(R1)+   ;402=100001
3089  011272  012711  100002              MOV     #100002,(R1)    ;404=100002
3090  011276  005741                      TST     -(R1)           ;R1=402
3091  011300  132421                      BITB    (R4)+,(R1)+     ;**ANDED RESULT= 000
3092  011302  100401                      BMI     1$              ;ERROR IF MINUS
3093  011304  001401                      BEQ     2$              ;BRANCH IF GOOD
3094  011306  104001              1$:   ERROR   ^1              ;CPU ERROR
3095                                                          ;CC SHOULD = 0100
```

```
                        ************ DOUBLE OPERAND TESTS ****************                SEQ 0059

 3096  011310  132124              2$:     BITB    (R1)+,(R4)+     ;** ANDED RESULT = 200
 3097  011312  001401                      BEQ     3$              ;ERROR IF EQUAL
 3098  011314  100401                      BMI     4$              ;BRANCH IF GOOD
 3099  011316  104001              3$:     ERROR   +1              ;CPU ERROR
 3100                                                ;CC SHOULD= 1000 R4=402 R1=404
 3101  011320  022421              4$:     CMP     (R4)+,(R1)+     ;RESULT =+1
 3102  011322  001402                      BEQ     5$              ;ERROR IF EQUAL
 3103  011324  103001                      BCC     5$              ;ERROR IF NO CARRY
 3104  011326  100401                      BMI     6$              ;BRANCH IF GOOD
 3105  011330  104001              5$:     ERROR   +1              ;CPU ERROR
 3106                                                ;CC SHOULD = 0000
 3107  011332  005341              6$:     DEC     -(R1)           ;404=100001
 3108  011334  005741                      TST     -(R1)           ;R4=404 R1=402
 3109  011336  022124                      CMP     (R1)+,(R4)+     ;RESULT =0
 3110  011340  001401                      BEQ     8$              ;BRANCH IF GOOD
 3111                                                ;CC SHOULD = 0100 R1=404 R4=406
 3112  011342  104001              7$:     ERROR   +1              ;CPU ERROR
 3113  011344                      8$:
 3114
 3115                              ;
 3116  011344                      MS33:
 3117
 3118                              ;       TEST SUB MODE 3,3
 3119  011344  005004                      CLR     R4              ;R4=0
 3120  011346  012701  000002              MOV     #2,R1           ;R1=2
 3121  011352  012702  000400              MOV     #400,R2         ;R2=400
 3122  011356  012714  000400              MOV     #400,(R4)       ;0=400
 3123  011362  012711  000402              MOV     #402,(R1)       ;2=402
 3124  011366  012722  000200              MOV     #200,(R2)+      ;400=200
 3125  011372  012712  054320              MOV     #54320,(R2)     ;402=54320
 3126  011376  163431                      SUB     @(R4)+,@(R1)+   ;54320 - 200=54120
 3127  011400  001402                      BEQ     1$              ;ERROR IF ZERO
 3128  011402  103401                      BCS     1$              ;ERROR IF CARRY
 3129  011404  100001                      BPL     2$              ;BRANCH IF GOOD
 3130  011406  104001              1$:     ERROR   +1              ;CPU ERROR
 3131                                                ;CC SHOULD =0001
 3132  011410  022712  054120      2$:     CMP     #54120,(R2)     ; TEST R4 AUTO-INC AND RESULT
 3133  011414  001401                      BEQ     4$              ;BRANCH IF GOOD
 3134  011416  104001              3$:     ERROR   +1              ;CPU ERROR
 3135                                                ;CC SHOULD = 0100 R4=2 R1=4
 3136  011420  005067  166354      4$:     CLR     0               ;RESTORE VECTORS
 3137  011424  005067  166352              CLR     2               ;       "
 3138
 3139
 3140                              ;
 3141  011430                      MCB44:
 3142
 3143                              ;       TEST CMP, BIT MODE 4,4
 3144  011430  012704  000400              MOV     #400,R4                 ;R4=400
 3145  011434  012701  000402              MOV     #402,R1                 ;R1=402
 3146  011440  012721  125366              MOV     #125366,(R1)+           ;402=125366 R1=404
 3147  011444  012724  173001              MOV     #173001,(R4)+           ;400=173001 R4=402
 3148  011450  024441                      CMP     -(R4),-(R1)             ;173001 - 125366=045603 CC=0000
 3149  011452  103401                      BCS     1$                      ;ERROR IF CARRY
 3150  011454  100001                      BPL     2$                      ;BRANCH IF GOOD
 3151                                                                      ;BAD COMPARE
```

```
                        ************ DOUBLE OPERAND TESTS *****************                    SEQ 0060

 3152  011456  104001                1$:     ERROR   .1              ;CPU ERROR
 3153  011460  005204                2$:     INC     R4
 3154  011462  005201                        INC     R1                      ;R1=403
 3155  011464  000261                        SEC                             ;SET CARRY
 3156  011466  134144                        BITB    -(R1),-(R4)             ;173^1=0
 3157  011470  103001                        BCC     3$                      ;C SHOULD REMAIN SET
 3158  011472  001401                        BEQ     4$                      ;BRANCH IF GOOD
 3159                                                                        ;INCORRECT COMPARE R4=400 R1=402
 3160  011474  104001                3$:     ERROR   .1              ;CPU ERROR
 3161  011476  005724                4$:     TST     (R4)+                   ;R4=402
 3162  011500  005201                        INC     R1                      ;R1=403
 3163  011502  124441                        CMPB    -(R4),-(R1)             ;173 - 173=0
 3164  011504  001401                        BEQ     6$                      ;BRANCH IF GOOD
 3165                                                                        ;BAD COMPARE
 3166  011506  104001                5$:     ERROR   .1              ;CPU ERROR
 3167  011510                        6$:
 3168                                ;
 3169  011510                        MA55:
 3170
 3171                                ;       TEST ADD MODE 5,5
 3172  011510  012704  000400                MOV     #400,R4
 3173  011514  012724  000001                MOV     #1,(R4)+                ;400=1
 3174  011520  012724  177776                MOV     #-2,(R4)+               ;402=-2
 3175  011524  012724  000400                MOV     #400,(R4)+              ;404=400
 3176  011530  012714  000402                MOV     #402,(R4)               ;406=402 R4=406
 3177  011534  012701  000410                MOV     #410,R1                 ;R1=410
 3178  011540  065451                        ADD     @-(R4),@-(R1)           ;1+ -2= -1
 3179  011542  001402                        BEQ     1$                      ;ERROR IF ZERO
 3180  011544  100001                        BPL     1$                      ;ERROR IF PLUS
 3181  011546  103001                        BCC     2$                      ;BRANCH IF GOOD
 3182  011550  104001                1$:     ERROR   .1              ;CPU ERROR
 3183                                                                        ;BAD ADU
 3184  011552  062704  000004        2$:     ADD     #4,R4                   ;R4=410
 3185  011556  065154                        ADD     @-(R1),@-(R4)           ;-1 + 1 = 0
 3186  011560  001401                        BEQ     4$                      ;BRANCH IF GOOD
 3187  011562  104001                3$:     ERROR   .1              ;CPU ERROR
 3188                                                                        ;CC SHOULD= 0100 R4=406 R1=402
 3189  011564                        4$:
 3190
 3191                                ;
 3192                                ;
 3193                                ;
 3194                                ;-----------------------------------
 3195                                ;TEST DOP BIT(B) MODE 6,6
 3196                                ;
 3197                                        ;
 3198                                        ;
 3199                                        ;
 3200  011564                        MB66:
 3201
 3202                                ;       TEST BIT, BITB MODE 6,6
 3203  011564  005004                        CLR     R4                      ;R4=0
 3204  011566  012701  000400                MOV     #400,R1                 ;
 3205  011572  012721  125252                MOV     #125252,(R1)+           ;400=125252
 3206  011576  012721  000001                MOV     #1,(R1)+                ;402=1
 3207  011602  012721  100000                MOV     #100000,(R1)+           ;404=1000000  R1=406
```

```
3208  011606  036461  000400  177774          BIT     400(R4),-4(R1)          ;(400)&(402)=0
3209  011614  001401                          BEQ     2$                      ;BRANCH IF GOOD
3210                                                                          ;CC SHOULD = 0100
3211  011616  104001                  1$:     ERROR   .1                  ;CPU ERROR
3212  011620  136461  000405  177772  2$:     BITB    405(R4),-6(R1)          ;(405)&(400)=200
3213  011626  001401                          BEQ     3$                      ;ERROR IF ZERO
3214  011630  100401                          BMI     4$                      ;BRANCH IF GOOD
3215                                                                          ;CC SHOULD = 1000
3216  011632  104001                  3$:     ERROR   .1                  ;CPU ERROR
3217  011634                          4$:
3218
3219                                  ;
3220  011634                          MS77:
3221
3222                                  ;       TEST SUB MODE 7,7
3223  011634  012704  000400                  MOV     #400,R4         ;
3224  011640  005001                          CLR     R1
3225  011642  012724  177776                  MOV     #-2,(R4)+               ;400=-2
3226  011646  012724  177777                  MOV     #-1,(R4)+               ;402=-1
3227  011652  012724  000400                  MOV     #400,(R4)+              ;404=400 R4=406
3228  011656  012711  000402                  MOV     #402,(R1)               ;0=402
3229  011662  005201                          INC     R1                      ;R1=1
3230  011664  167471  177372  000403          SUB     @-406(R4),@403(R1)      ;-2 - -1 = -1
3231  011672  001401                          BEQ     1$                      ;ERROR  IF ZERO
3232  011674  100401                          BMI     2$                      ;BRANCH IF GOOD
3233                                                                          ;CC SHOULD=1000
3234  011676  104001                  1$:     ERROR   .1                  ;CPU ERROR
3235  011700  167174  177777  177776  2$:     SUB     @-1(R1),@-2(R4)         ;-1 - -1 = 0
3236  011706  001401                          BEQ     4$                      ;BRANCH IF GOOD
3237  011710  104001                  3$:     ERROR   .1                  ;CPU ERROR
3238                                                                          ;ERROR ON SUBTRACT 400=0
3239  011712  005067  166062          4$:     CLR     0                       ;RESTORE VECTORS
3240  011716  005067  166060                  CLR     2                       ;       "
3241
3242
3243
3244  011722                          MRLO:
3245
3246                                  ;       TEST ROL, ROLB MODE 0
3247  011722  012704  125252                  MOV     #125252,R4              ;R4=125252
3248  011726  000277                          SCC                             ;CC=1111
3249  011730  006104                          ROL     R4                      ;R4=052525 WITH CARRY SET
3250  011732  102004                          BVC     1$                      ;ERROR IF V CLEAR
3251  011734  103003                          BCC     1$                      ;ERROR IF CARRY CLEAR
3252  011736  022704  052525                  CMP     #052525,R4              ;SEE IF R0 = EXPECTED
3253  011742  001401                          BEQ     2$                      ;ERROR IF R4 NE EXPECTD
3254  011744  104001                  1$:     ERROR   .1                  ;CPU ERROR
3255                                                                      ;ROL FAILED, CC SHOULD=0011
3256  011746  012704  125252          2$:     MOV     #125252,R4              ;R4=125252
3257  011752  000257                          CCC                             ;CC=0000
3258  011754  106104                          ROLB    R4                      ;ROTATE EVEN BYTE
3259  011756  103005                          BCC     3$                      ;ERROR IF NO CARRY
3260  011760  102004                          BVC     3$                      ;ERROR IF NO OVERFLOW
3261  011762  100403                          BMI     3$                      ;ERROR IF MINUS
3262  011764  022704  125124                  CMP     #125124,R4              ;SEE IF R4 = EXPECTED
3263  011770  001401                          BEQ     4$                      ;BRANCH IF GOOD
```

```
3264  011772  104001                  3$:     ERROR   ;1              ;CPU ERROR
3265                                                                  ;ROLB FAILED, CC SHOULD=1011, R4=125125
3266  011774                          4$:
3267
3268                                  ;
3269  011774                          MRLB1:
3270
3271                                  ;       TEST ROL, ROLB MODE 1
3272  011774  005004                          CLR     R4                      ;R4=0
3273  011776  012714  052525                 MOV     #52525,(R4)             ;0=52525
3274  012002  006114                          ROL     (R4)                    ;**TEST INSTRUCTION, 0=125252
3275  012004  100005                          BPL     1$                      ;ERROR IF PLUS
3276  012006  102004                          BVC     1$                      ;ERROR IF NO OVERFLOW
3277  012010  103403                          BCS     1$                      ;ERROR IF CARRY
3278  012012  021427  125252                 CMP     (R4),#125252            ;SEE IF R4=EXPECTED
3279  012016  001401                          BEQ     2$                      ;BRANCH IF GOOD
3280                                                                        ;BAD ROL ,CC SHOULD=1010
3281  012020  104001                  1$:     ERROR   ;1              ;CPU ERROR
3282  012022  012714  125252         2$:     MOV     #125252,(R4)            ;0=125252
3283  012026  005204                          INC     R4                      ;R4=1
3284  012030  000277                          SCC                             ;CC=1111
3285  012032  106114                          ROLB    (R4)                    ;**TEST INSTRUCTION
3286  012034  100406                          BMI     3$                      ;ERROR IF RESULT IS POSITIVE
3287  012036  103005                          BCC     3$                      ;ERROR IF NO CARRY
3288  012040  102004                          BVC     3$                      ;ERROR IF V CLEAR
3289  012042  005304                          DEC     R4                      ;R4=0
3290  012044  022714  052652                 CMP     #52652,(R4)             ;ERROR IF 0 NE EXPECTED
3291  012050  001401                          BEQ     4$                      ;BRANCH IF GOOD
3292  012052  104001                  3$:     ERROR   ;1              ;CPU ERROR
3293                                                                  ;BAD ROLB ODD BYTE,CC SHOULD=1011
3294  012054                          4$:
3295
3296                                  ;
3297  012054                          MRL2:
3298
3299                                  ;       TEST ROL, ROLB MODE 2
3300  012054  005004                          CLR     R4                      ;R4=0
3301  012056  012714  100000                 MOV     #100000,(R4)            ;0=100000
3302  012062  000257                          CCC                             ;CC=0000
3303  012064  006124                          ROL     (R4)+                   ;*TEST INSTRUCTION
3304  012066  103002                          BCC     1$                      ;ERROR IF NO CARRY
3305  012070  102001                          BVC     1$                      ;ERROR IF NO OVERFLOW
3306  012072  001401                          BEQ     2$                      ;BRANCH IF GOOD
3307                                                                        ;ROL FAILED ,CCSHOULD= 0100
3308  012074  104001                  1$:     ERROR   ;1              ;CPU ERROR
3309  012076  005304                  2$:     DEC     R4
3310  012100  005304                          DEC     R4
3311  012102  001012                          BNE     3$                      ;ERROR IN AUTO-DEC
3312  012104  012714  004040                 MOV     #4040,(R4)              ;0=4040
3313  012110  000241                          CLC
3314  012112  106124                          ROLB    (R4)+                   ;**TEST INSTURCTION
3315  012114  103405                          BCS     3$                      ;ERROR IF CARRY SET
3316  012116  102404                          BVS     3$                      ;ERROR IF V
3317  012120  005304                          DEC     R4
3318  012122  022714  004100                 CMP     #04100,(R4)             ;SEE IF 0= EXPECTED RESULT
3319  012126  001401                          BEQ     4$                      ;BRANCH IF GOOD
```

```
3320  012130  104001                  3$:     ERROR   ;1              ;CPU ERROR
3321                                                                  ;BAD ROL
3322  012132                          4$:
3323
3324                                  ;
3325  012132                          MRL3:
3326
3327                                  ;       TEST ROL, ROLB MODE 3
3328  012132  005004                          CLR     R4                      ;R4=0
3329  012134  012714  052525                  MOV     #052525,(R4)                    ;0=52525
3330  012140  000277                          SCC                             ;CC=1111
3331  012142  006137  000000                  ROL     @#0                     ;**TEST INSRUCTION MODE 3 WITH PC
3332  012146  100005                          BPL     1$                      ;ERROR IF PLUS
3333  012150  102004                          BVC     1$                      ;ERROR IF NO OVERFLOW
3334  012152  103403                          BCS     1$                      ;ERROR IF CARRY
3335  012154  022714  125253                  CMP     #125253,(R4)            ;COMPARE RESULT WITH EXPECTED
3336  012160  001401                          BEQ     2$                      ;BRANCH IF GOOD
3337                                                                  ;BAD ROL CC SHOULD=1010
3338  012162  104001                  1$:     ERROR   ;1              ;CPU ERROR
3339  012164  012714  125252          2$:     MOV     #125252,(R4)            ;0=125252
3340  012170  000261                          SEC                             ;CC=---1
3341  012172  106137  000000                  ROLB    @#0                     ;**TEST INSTRUCTION
3342  012176  100402                          BMI     3$                      ;ERROR IF MINUS
3343  012200  103001                          BCC     3$                      ;ERROR IF NO CARRY
3344  012202  102401                          BVS     4$                      ;BRANCH IF OVERFLOW
3345  012204  104001                  3$:     ERROR   ;1              ;CPU ERROR
3346                                                                  ;BAD ROL, CC SHOULD=1011
3347  012206                          4$:
3348
3349                                  ;
3350  012206                          MRL4:
3351
3352                                  ;       TEST ROL MODE 4
3353  012206  005001                          CLR     R1                      ;R1=0
3354  012210  012704  000002                  MOV     #2,R4                   ;R4=2
3355  012214  012711  054321                  MOV     #54321,(R1)             ;0=54321
3356  012220  000277                          SCC                             ;CC=1111
3357  012222  006144                          ROL     -(R4)                   ;**TEST INSTRUCTION
3358  012224  100007                          BPL     1$                      ;ERROR IF PLUS
3359  012226  102006                          BVC     1$                      ;ERROR IF NO OVERFLOW
3360  012230  103405                          BCS     1$                      ;ERROR IF CARRY
3361  012232  022711  130643                  CMP     #130643,(R1)            ;SEE IF EXPECTED RESULT
3362  012236  001002                          BNE     1$                      ;BRANCH IF ROL FAILED
3363  012240  005704                          TST     R4                      ;SEE IF AUTO-DEC WORKED
3364  012242  001401                          BEQ     2$                      ;BRANCH IF GOOD
3365                                                                  ;ERROR! BAD ROL INST
3366  012244  104001                  1$:     ERROR   ;1              ;CPU ERROR
3367  012246                          2$:
3368
3369                                  ;
3370  012246                          MMRL5:
3371
3372                                  ;       TEST ROL MODE 5
3373  012246  005004                          CLR     R4                      ;R4=0
3374  012250  012714  000400                  MOV     #400,(R4)               ;0=400
3375  012254  012734  123456                  MOV     #123456,@(R4)+          ;400=123465, R4=2
```

```
3376  012260  000277                          SCC                             ;CC=1111
3377  012262  006154                          ROL     @-(R4)                  ;**TEST INSTRUCTION
3378  012264  100410                          BMI     1$                      ;ERROR IF RESULT IS MINUS
3379  012266  103007                          BCC     1$                      ;ERROR IF NO CARRY
3380  012270  102006                          BVC     1$                      ;ERROR IF NO OVERFLOW
3381  012272  005704                          TST     R4                      ;SEE IF AUTO-DEC WORKED
3382  012274  001004                          BNE     1$                      ;ERROR OF AUTO-DEC
3383  012276  022737  047135  000400          CMP     #47135,@#400            ;SEE  IF CORRECT RESULT
3384  012304  001401                          BEQ     2$                      ;BRANCH IF GOOD
3385                                                                          ;BAD ROL MODE 5
3386  012306  104001                  1$:     ERROR   +1              ;CPU ERROR
3387  012310                          2$:
3388
3389                                  ;
3390  012310                          MRL6:
3391
3392                                  ;       TEST ROL MODE 6
3393  012310  012704  000400                  MOV     #400,R4                 ;R4=400
3394  012314  005001                          CLR     R1                      ;R1=0
3395  012316  012711  032525                  MOV     #32525,(R1)             ;0=32525
3396  012322  000277                          SCC
3397  012324  006164  177400                  ROL     -400(R4)                ;**TEST INSTRUCTION
3398  012330  100405                          BMI     1$                      ;ERROR IF MINUS
3399  012332  103404                          BCS     1$                      ;ERROR IF CARRY
3400  012334  102403                          BVS     1$                      ;ERROR IF OVERFLOW
3401  012336  022711  065253                  CMP     #65253,(R1)             ;SEE IF CORRECT RESULT
3402  012342  001401                          BEQ     2$                      ;BRANCH IF GOOD
3403                                                                          ;BAD ROL MODE 6
3404  012344  104001                  1$:     ERROR   +1              ;CPU ERROR
3405  012346                          2$:
3406
3407                                  ;
3408  012346                          MRL7:
3409
3410                                  ;       TEST ROL MODE 7
3411  012346  012704  000400                  MOV     #400,R4                 ;R4=400
3412  012352  005037  000402                  CLR     @#402                   ;402=0
3413  012356  012737  100000  000000          MOV     #100000,@#0             ;0=100000
3414  012364  006174  000002                  ROL     2(R4)                   ;**TEST INSTRUCTION
3415  012370  100406                          BMI     1$                      ;ERROR IF MINUS
3416  012372  001005                          BNE     1$                      ;ERROR IF NOT ZERO
3417  012374  103004                          BCC     1$                      ;ERROR IF NO CARRY
3418  012376  102003                          BVC     1$                      ;ERROR IF NO OVERFLOW
3419  012400  005737  000000                  TST     @#0                     ;CHECK RESULT
3420  012404  001401                          BEQ     2$                      ;BRANCH IF GOOD
3421                                                                          ;BAD ROL MODE 7
3422  012406  104001                  1$:     ERROR   +1              ;CPU ERROR
3423  012410                          2$:
3424
3425                                  ;
3426  012410                          MSW37:
3427
3428                                  ;       TEST SWAB MODE 37
3429  012410  012704  000400                  MOV     #400,R4                 ;R4=400
3430  012414  012714  040700                  MOV     #40700,(R4)             ;400= 101 300
3431  012420  000337  000400                  SWAB    @#400                   ;400 SHOULD = 300 101
```

```
3432  012424  100406                          BMI     1$                      ;ERROR IF MINUS
3433  012426  022714  140101                  CMP     #140101,(R4)            ;SEE IF EXPECTED RESULT
3434  012432  001003                          BNE     1$                      ;BRANCH IF BAD
3435  012434  000337  000400                  SWAB    @#400                   ;400=101 300
3436  012440  100401                          BMI     2$                      ;BRANCH IF GOOD
3437                                                                          ;ERROR! BAD SWAB MODE 37
3438  012442  104001                  1$:     ERROR   +1              ;CPU ERROR
3439  012444                          2$:
3440
3441                                  ;
3442  012444                          MRR0:
3443
3444                                  ;       TEST ROR MODE 0
3445  012444  012704  052525                  MOV     #52525,R4               ;R4=52525
3446  012450  000257                          CCC                             ;CC=0000
3447  012452  006004                          ROR     R4                      ;R4 SHOULD = 25252 WITH CARRY
3448  012454  103003                          BCC     1$                      ;ERROR IF NO CARRY
3449  012456  022704  025252                  CMP     #25252,R4               ;SEE IF R4= EXPECTED
3450  012462  001401                          BEQ     2$                      ;BRANCH IF GOOD
3451                                                                          ;ROR MODE 0 FAILED
3452  012464  104001                  1$:     ERROR   +1              ;CPU ERROR
3453  012466                          2$:
3454
3455                                  ;
3456  012466                          MRRB1:
3457
3458                                  ;       TEST RORB MODE 1
3459  012466  005004                          CLR     R4                      ;R4=0
3460  012470  012714  000001                  MOV     #1,(R4)                 ;0=1
3461  012474  000277                          SCC                             ;CC=1111
3462  012476  106014                          RORB    (R4)                    ;0=000200, NO C
3463  012500  103004                          BCC     1$                      ;ERROR IF NO CARRY
3464  012502  100003                          BPL     1$                      ;ERROR IF PLUS
3465  012504  022714  000200                  CMP     #200,(R4)               ;CHECK RESULT
3466  012510  001401                          BEQ     2$                      ;BRANCH IF GOOD
3467  012512  104001                  1$:     ERROR   +1              ;CPU ERROR
3468                                                                  ;BAD RESULT
3469  012514                          2$:
3470
3471                                  ;
3472
3473  012514                          MJ:
3474
3475                                  ;       TEST JMP - ALL MODES
3476  012514  012737  000001  003032          MOV     #1,@#SEQ                ;SETUP TEST SEQUENCER
3477  012522  012701  012566                  MOV     #MJU1,R1                ;SET MODE 1 JUMP ADDRESS
3478  012526  000111                          JMP     (R1)                    ;=JMP MODE 1
3479  012530  023727  003032  000002  MJU2:   CMP     @#SEQ,#2                ;CHECK FOR CORRECT SEQUENCE
3480  012536  001401                          BEQ     MJU2A                   ;BRANCH IF GOOD
3481                                                                          ;MODE 2 JUMP FAILED
3482  012540  104001                  1$:     ERROR   +1              ;CPU ERROR
3483  012542  020127  012532          MJU2A:  CMP     R1,#MJU2+2              ;CHECK FOR AUTO-INCREMENT
3484  012546  001401                          BEQ     MJU2B                   ;BRANCH IF GOOD
3485  012550  104001                  2$:     ERROR   +1              ;CPU ERROR
3486                                                                  ;AUTO-INCR FAILED
3487  012552  005237  003032          MJU2B:  INC     @#SEQ                   ;UPDATE TEST SEQUENCER
```

```
3488  012556  012701  012564                MOV     @MJ2,R1                 ;SETUP MODE 3 JUMP
3489  012562  000131                       JMP     @(R1)+                  ;JUMP MODE 3
3490  012564  012612                MJ2:   .WORD   MJU3                    ;MODED 3 DETINATION
3491  012566  023727  003032  000001  MJU1:  CMP     @#SEQ,#1                ; TEST FOR CORRECT SEQUENCE
3492  012574  001401                       BEQ     MJU1A                   ;BRANCH IF GOOD
3493  012576  104001                3$:    ERROR   ;1              ;CPU ERROR
3494                                                                 ;JMP OUT OF SEQUENCE
3495  012600  005237  003032        MJU1A: INC     @#SEQ                   ;UPDATE SEQUENCE
3496  012604  012701  012530              MOV     @MJU2,R1                ;SETUP MODE 2 DESTINATION
3497  012610  000121                       JMP     (R1)+                   ;JUMP MODE 2
3498  012612  023727  003032  000003  MJU3:  CMP     @#SEQ,#3                ; TEST FOR CORRECT SEQUENCE
3499  012620  001401                       BEQ     MJU3A                   ;BRANCH IF GOOD
3500  012622  104001                4$:    ERROR   ;1              ;CPU ERROR
3501                                                                 ;JMP OUT OF SEQUENCE
3502  012624  022701  012566        MJU3A: CMP     @MJ2+2,R1               ; TEST AUTO-INCREMENT
3503  012630  001401                       BEQ     MJU3B                   ;BRANCH IF GOOD
3504  012632  104001                5$:    ERROR   ;1              ;CPU ERROR
3505                                                                 ;AUTO-INCREMENT FAILED MODE 3
3506  012634  005237  003032        MJU3B: INC     @#SEQ                   ;UPDATE SEQUENCER
3507  012640  012701  012710              MOV     @MJU4+2,R1              ;SETUP DESTINATION MODE 4
3508  012644  000141                       JMP     -(R1)                   ;EXECUTE JUMP MODE 4
3509  012646  000000                MJU5:  HALT
3510  012650  022701  012744              CMP     @MJ5,R1                 ;CHECK AUTO-DECREMENT
3511  012654  001401                       BEQ     MJU5A                   ;BRANCH IF GOOD AUTO-DEC
3512  012656  104001                6$:    ERROR   ;1              ;CPU ERROR
3513                                                                 ;AUTO-DEC FAIELD MODE 5
3514  012660  023727  003032  000005  MJU5A: CMP     @#SEQ,#5                ; TEST CORRECT SEQUENCE
3515  012666  001401                       BEQ     MJU5B                   ;BRANCH IF GOOD SEQUENCE
3516  012670  104001                7$:    ERROR   ;1              ;CPU ERROR
3517                                                                 ;JMP OUT OF SEQUENCE
3518  012672  005237  003032        MJU5B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNT
3519  012676  012701  012741              MOV     @MJU6-5,R1              ;SETUP DESTINATION MODE6
3520  012702  000161  000005              JMP     +5(R1)                  ;JUMP MODE 6
3521                                ;;
3522  012706  000240                MJU4:  NOP
3523  012710  022701  012706              CMP     @MJU4,R1                ; TEST AUTO-DECR
3524  012714  001401                       BEQ     MJU4A                   ;BRANCH IF GOOD
3525  012716  104001                8$:    ERROR   ;1              ;CPU ERROR
3526                                                                 ;MODE 4 AUTO-DEC FAILED
3527  012720  023727  003032  000004  MJU4A: CMP     @#SEQ,#4                ; TEST FOR CORRECT SEQUENCE
3528  012726  001401                       BEQ     MJU4B                   ;BRANCH IF CORRECT SEQUENCE
3529  012730  104001                9$:    ERROR   ;1              ;CPU ERROR
3530                                                                 ;INCORRECT JMP SEQUENCE
3531  012732  005237  003032        MJU4B: INC     @#SEQ                   ;UPDATE SEQUENCE
3532  012736  012701  012746              MOV     @MJ5+2,R1               ;SETUP MODE 5 POINTER
3533  012742  000151                       JMP     @-(R1)                  ;EXECUTE MODE 5 JMP
3534                                ;
3535  012744  012650                MJ5:   .WORD   MJU5+2                  ;POINTER MODE 5
3536                                ;
3537  012746  022737  000006  003032  MJU6:  CMP     #6,@#SEQ                ;CHECK FOR CORRECT SEQUENCE
3538  012754  001401                       BEQ     MJU6A                   ;BRANCH IF GOOD
3539  012756  104001                10$:   ERROR   ;1              ;CPU ERROR
3540                                                                 ;INCORRECT SEQUENCE
3541  012760  005237  003032        MJU6A: INC     @#SEQ                   ;UPDATE SEQUENCER
3542  012764  012701  013004              MOV     @MJ7+10,R1              ;SETUP INDEX
3543  012770  000171  177770              JMP     @-10(R1)                ;EXECUTE MODE 7 JUMP
```

C6,


```
3544  012774  013000                  MJ7:    .WORD   MJU7                    ;POINTER FOR MODE 7
3545  012776  000000                          HALT
3546  013000  022737  000007  003032  MJU7:   CMP     #7,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3547  013006  001401                          BEQ     MJU7E                   ;BRANCH IF GOOD SEQUENCE
3548  013010  104001                  11$:    ERROR   .1              ;CPU ERROR
3549                                                                  ; TESTING OUT OF SEQUENCE
3550  013012                          MJU7E:
3551                                  ;
3552
3553                                  ;       TEST THAT PRE-FETCH BUFFER CAN BE OVER WRITTEN
3554  013012  012701  013322                  MOV     #12$,R1                 ;SET UP R1 WITH ADDRESS OF ERROR
3555                                                                          ;ROUTINE
3556  013016  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3557  013020  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3558  013022  000111                  64$:    .WORD   111                     ;JMP    (R1)
3559  013024  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3560  013026  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3561  013030  000111                  65$:    .WORD   111                     ;JMP    (R1)
3562  013032  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3563  013034  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3564  013036  000111                  66$:    .WORD   111                     ;JMP    (R1)
3565  013040  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3566  013042  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3567  013044  000111                  67$:    .WORD   111                     ;JMP    (R1)
3568  013046  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3569  013050  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3570  013052  000111                  68$:    .WORD   111                     ;JMP    (R1)
3571  013054  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3572  013056  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3573  013060  000111                  69$:    .WORD   111                     ;JMP    (R1)
3574  013062  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3575  013064  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3576  013066  000111                  70$:    .WORD   111                     ;JMP    (R1)
3577  013070  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3578  013072  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3579  013074  000111                  71$:    .WORD   111                     ;JMP    (R1)
3580  013076  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3581  013100  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3582  013102  000111                  72$:    .WORD   111                     ;JMP    (R1)
3583  013104  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3584  013106  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3585  013110  000111                  73$:    .WORD   111                     ;JMP    (R1)
3586  013112  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3587  013114  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3588  013116  000111                  74$:    .WORD   111                     ;JMP    (R1)
3589  013120  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3590  013122  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3591  013124  000111                  75$:    .WORD   111                     ;JMP    (R1)
3592  013126  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3593  013130  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3594  013132  000111                  76$:    .WORD   111                     ;JMP    (R1)
3595  013134  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3596  013136  000240                          .WORD   NOP                     ;NOP INSTRUCTION
3597  013140  000111                  77$:    .WORD   111                     ;JMP    (R1)
3598  013142  012717                          MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
3599  013144  000240                          .WORD   NOP                     ;NOP INSTRUCTION
```

D6,


```
3600  013146  000111                  78$:    .WORD   111             ;JMP    (R1)
3601  013150  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3602  013152  000240                          .WORD   NOP             ;NOP INSTRUCTION
3603  013154  000111                  79$:    .WORD   111             ;JMP    (R1)
3604  013156  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3605  013160  000240                          .WORD   NOP             ;NOP INSTRUCTION
3606  013162  000111                  80$:    .WORD   111             ;JMP    (R1)
3607  013164  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3608  013166  000240                          .WORD   NOP             ;NOP INSTRUCTION
3609  013170  000111                  81$:    .WORD   111             ;JMP    (R1)
3610  013172  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3611  013174  000240                          .WORD   NOP             ;NOP INSTRUCTION
3612  013176  000111                  82$:    .WORD   111             ;JMP    (R1)
3613  013200  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3614  013202  000240                          .WORD   NOP             ;NOP INSTRUCTION
3615  013204  000111                  83$:    .WORD   111             ;JMP    (R1)
3616  013206  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3617  013210  000240                          .WORD   NOP             ;NOP INSTRUCTION
3618  013212  000111                  84$:    .WORD   111             ;JMP    (R1)
3619  013214  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3620  013216  000240                          .WORD   NOP             ;NOP INSTRUCTION
3621  013220  000111                  85$:    .WORD   111             ;JMP    (R1)
3622  013222  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3623  013224  000240                          .WORD   NOP             ;NOP INSTRUCTION
3624  013226  000111                  86$:    .WORD   111             ;JMP    (R1)
3625  013230  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3626  013232  000240                          .WORD   NOP             ;NOP INSTRUCTION
3627  013234  000111                  87$:    .WORD   111             ;JMP    (R1)
3628  013236  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3629  013240  000240                          .WORD   NOP             ;NOP INSTRUCTION
3630  013242  000111                  88$:    .WORD   111             ;JMP    (R1)
3631  013244  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3632  013246  000240                          .WORD   NOP             ;NOP INSTRUCTION
3633  013250  000111                  89$:    .WORD   111             ;JMP    (R1)
3634  013252  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3635  013254  000240                          .WORD   NOP             ;NOP INSTRUCTION
3636  013256  000111                  90$:    .WORD   111             ;JMP    (R1)
3637  013260  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3638  013262  000240                          .WORD   NOP             ;NOP INSTRUCTION
3639  013264  000111                  91$:    .WORD   111             ;JMP    (R1)
3640  013266  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3641  013270  000240                          .WORD   NOP             ;NOP INSTRUCTION
3642  013272  000111                  92$:    .WORD   111             ;JMP    (R1)
3643  013274  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3644  013276  000240                          .WORD   NOP             ;NOP INSTRUCTION
3645  013300  000111                  93$:    .WORD   111             ;JMP    (R1)
3646  013302  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3647  013304  000240                          .WORD   NOP             ;NOP INSTRUCTION
3648  013306  000111                  94$:    .WORD   111             ;JMP    (R1)
3649  013310  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
3650  013312  000240                          .WORD   NOP             ;NOP INSTRUCTION
3651  013314  000111                  95$:    .WORD   111             ;JMP    (R1)
3652  013316  000137  013324                  JMP     @#1294          ;JUMP OVER ERROR CALL
3653  013322                          128$:                           ;ERROR! PRE-FETCH BUFFER WAS NOT
3654                                                                  ;OVER WRITTEN
3655  013322  104001                          ERROR   +1              ;CPU ERROR
```

```
3656                                    ;
3657                                    ;       NOW RESTORE THE OVER WRITTEN JMP INSTRUCTIONS FOR THE NEXT PASS.
3658                                    ;
3659  013324  012702  000040            129$:   MOV     #32.,R2                 ;SET UP R2 AS COUNTER
3660  013330  012703  013022                    MOV     #64$,R3                 ;SET UP R3 AS POINTER
3661  013334  012713  000111            130$:   MOV     #111,(R3)               ;RESTORE OVER WRITTEN JUMPS
3662  013340  062703  000006                    ADD     #6,R3                   ;POINT TO NEXT OVER WRITTEN ADDR.
3663  013344  077205                            SOB     R2,130$                 ;DO UNTIL R2=0
3664
3665                                    ;
3666  013346                            MJP:
3667
3668                                    ;       TEST JMP MODES 17,27,37,67,77
3669  013346  012737  000000  003032            MOV     #0,@#SEQ                ;SETUP TEST SEQUNCER
3670  013354  000117                            JMP     (R7)                    ;JUMP MODE 17(SHOULD BE IN-LINE)
3671                                    ;
3672  013356  005737  003032            MJP17:  TST     @#SEQ                   ;CHECK SEQUENCE
3673  013362  001401                            BEQ     2$                      ;BRANCH IF GOOD
3674                                                                            ;ERROR! BAD JUMP
3675  013364  104001                    1$:     ERROR   .+1             ;CPU ERROR
3676  013366  005237  003032            2$:     INC     @#SEQ                   ;UPDATE SEQUENCE NUMBER
3677  013372  000127  000401                    JMP     #401                    ;JUMP MODE 27
3678                                                                            ;(THE #401=BR UPDATED PC+2)
3679                                    ;
3680  013376  000000                            HALT                            ;
3681                                    ;
3682  013400  023727  003032  000001    MJP27:  CMP     @#SEQ,#1                ;CHECK IF CORRECT SEQUENCE
3683  013406  001401                            BEQ     MJP27A                  ;BRANCH IF IN SEQUENCE
3684  013410  104001                    1$:     ERROR   .+1             ;CPU ERROR
3685                                                                    ; TEST OUT OF SEQUENCE
3686  013412  005237  003032            MJP27A: INC     @#SEQ                   ;UPDATE SEQUENCER
3687  013416  000137  013464                    JMP     @#MJP37                 ;JUMP MODE 37
3688  013422  023727  003032  000003    MJP67:  CMP     @#SEQ,#3                ;CHECK FOR CORRECT SEQUENCE
3689  013430  001401                            BEQ     MJP67A                  ;BRANCH IF IN SEQUENCE
3690  013432  104001                    2$:     ERROR   .+1             ;CPU ERROR
3691                                                                    ; TEST OUT OF SEQUENCE
3692  013434  005237  003032            MJP67A: INC     @#SEQ                   ;UPDATE SEQUENCER
3693  013440  000257                            CCC                             ;INSURE ZBIT=0
3694  013442  000177  000002                    JMP     @#MJP67B                ;JUMP MODE 7
3695                                    ;
3696  013446  000000                            HALT                            ;
3697                                    ;
3698  013450  013452                    MJP67B: .WORD   MJP77                   ;
3699                                    ;
3700  013452  023727  003032  000004    MJP77:  CMP     @#SEQ,#4                ; TEST FOR CORRECT SEQUENCE
3701  013460  001412                            BEQ     MJP77E                  ;BRANCH IF IN SEQUENCE
3702  013462  104001                    3$:     ERROR   .+1             ;CPU ERROR
3703                                                                    ; TEST OUT OF SEQUENCE
3704  013464  023727  003032  000002    MJP37:  CMP     @#SEQ,#2                ; TEST FOR CORRECT SEQUENCE
3705  013472  001401                            BEQ     MJP37A                  ;BRANCH IF IN SEQUENCE
3706  013474  104001                    4$:     ERROR   .+1             ;CPU ERROR
3707                                                                    ; TEST OUT OF SEQUENCE
3708  013476  005237  003032            MJP37A: INC     @#SEQ                   ;UPDATE SEQUENCER
3709  013502  000167  177714                    JMP     MJP67                   ;JUMP MODE 6
3710
3711                                    ;
                                        ;
```

```
3712  013506                                MJP77E:
3713                                        ;
3714                                        ;
3715  013506                                MJSR:
3716
3717                                        ;       TEST JSR ALL MODES
3718  013506  010637  003034                       MOV     R6,@#SPS                ;SAVE STACK POINTER LOCATION
3719  013512  010637  003036                       MOV     R6,@#SPSJ               ;
3720  013516  162737  000002  003036               SUB     #2,@#SPSJ               ;SPSJ = R6 AFTER DECRIMENT
3721  013524  012737  000001  003032               MOV     #1,@#SEQ                ;SETUP SEQUENCE COUNTER
3722  013532  012701  013626                       MOV     #MJSR1,R1               ;SETUP INITIAL JUMP IN MODE 1
3723  013536  005004                               CLR     R4                      ;
3724  013540  005104                               COM     R4                      ;R4=-1 TO BE SAVED ON STACK
3725  013542  004411                               JSR     R4,(R1)                 ;JSR MODE 1
3726                                        ;
3727  013544  022737  000002  003032  MJSR2:  CMP     #2,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3728  013552  001401                               BEQ     MJSR2A                  ;BRANCH IF GOOD
3729  013554  104001                       5$:     ERROR   +1              ;CPU ERROR
3730                                                                        ;MODE 2 JUMPED TO OUT OF SEQUENCE
3731  013556  023706  003036          MJSR2A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3732  013562  001006                               BNE     6$                      ;BRANCH IF STACK INCORRECT
3733  013564  021627  125252                       CMP     (R6),#125252            ;VERIFY CONTENTS OF STACK
3734  013570  001003                               BNE     6$                      ;BRANCH IF DATA ON STACK INCORRECT
3735  013572  022704  013706                       CMP     #MJSR4,R4               ;SEE IF CORRECT RETURN ADDRESS
3736  013576  001401                               BEQ     MJSR2B                  ;BRANCH IF GOOD
3737  013600  104001                       6$:     ERROR   +1              ;CPU ERROR
3738                                                                        ;JSR MODE 2 FAILED
3739  013602  005237  003032          MJSR2B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3740  013606  013706  003034                       MOV     @#SPS,R6                ;RELOAD STACK POINTER
3741  013612  012701  013624                       MOV     #MJSRA,R1               ;SETUP JSR MODE 3
3742  013616  005004                               CLR     R4                      ;DIFFERENT DATA TO R4
3743  013620  004431                               JSR     R4,@(R1)+               ;JSR MODE 3
3744  013622  000000                               HALT
3745  013624  013772                       MJSRA:  .WORD   MJSR3                   ;LITERAL FOR JUMP MODE 3
3746                                        ;
3747  013626  022737  000001  003032  MJSR1:  CMP     #1,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3748  013634  001401                               BEQ     MJSR1A                  ;BRANCH IF GOOD
3749  013636  104001                       7$:     ERROR   +1              ;CPU ERROR
3750                                                                        ;MODE 1 JUMPED TO OUT OF SEQUENCE
3751  013640  023706  003036          MJSR1A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3752  013644  001006                               BNE     8$                      ;BRANCH IF STACK INCORRECT
3753  013646  021627  177777                       CMP     (R6),#-1                ;VERIFY CONTENT OF STACK
3754  013652  001003                               BNE     8$                      ;BRANCH IF DATA ON STACK INCORRECT
3755  013654  022704  013544                       CMP     #MJSR2,R4               ;SEE IF CORRECT RETURN ADDRESS
3756  013660  001401                               BEQ     MJSR1B                  ;BRANCH IF GOOD
3757  013662  104001                       8$:     ERROR   +1              ;CPU ERROR
3758                                                                        ;JSR MODE 2 FAILED
3759  013664  005237  003032          MJSR1B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3760  013670  013706  003034                       MOV     @#SPS,R6                ;RELOAD STACK POINTER
3761  013674  012704  125252                       MOV     #125252,R4              ;SETUP R4 DATA
3762  013700  012701  013544                       MOV     #MJSR2,R1               ;SETUP MODE 2 JUMP ADDRESS
3763  013704  004421                               JSR     R4,(R1)+                ;+JUMP MODE 2
3764                                        ;
3765                                        ;
3766  013706  022737  000004  003032  MJSR4:  CMP     #4,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3767  013714  001401                               BEQ     MJSR4A                  ;BRANCH IF GOOD
```

```
3768  013716  104001                  9$:     ERROR   ;1              ;CPU ERROR
3769                                                                  ;MODE 4 JUMPED TO OUT OF SEQUENCE
3770  013720  023706  003036          MJSR4A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3771  013724  001006                          BNE     10$                     ;BRANCH IF STACK INCORRECT
3772  013726  021627  052525                  CMP     (R6),#052525            ;VERIFY CONTENTS OF STAACK
3773  013732  001003                          BNE     10$                     ;BRANCH IF DATA ON STACK INCORRECT
3774  013734  022704  014054                  CMP     #MJSR6,R4               ;SEE IF CORRECT RETURN ADDRESS
3775  013740  001401                          BEQ     MJSR4B                  ;BRANCH IF GOOD
3776  013742  104001                  10$:    ERROR   ;1              ;CPU ERROR
3777                                                                  ;JSR MODE 4 FAILED
3778  013744  005237  003032          MJSR4B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3779  013750  013706  003034                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3780  013754  012704  000377                  MOV     #377,R4                 ;SETUP R4 DATA
3781  013760  012701  013772                  MOV     #MJSR8+2,R1             ;SETUP JSR VECTOR
3782  013764  004451                          JSR     R4,@-(R1)               ;JSR MODE 5
3783  013766  000000                          HALT
3784  013770  014140                  MJSR8:  .WORD   MJSR5                   ;MODE 5 VECTOR
3785                                   ;
3786                                   ;
3787  013772  022737  000003  003032  MJSR3:  CMP     #3,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3788  014000  001401                          BEQ     MJSR3A                  ;BRANCH IF GOOD
3789  014002  104001                  11$:    ERROR   ;1              ;CPU ERROR
3790                                                                  ;MODE 3 JUMPED TO OUT OF SEQUENCE
3791  014004  023706  003036          MJSR3A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3792  014010  001006                          BNE     12$                     ;BRANCH IF STACK INCORRECT
3793  014012  021627  000000                  CMP     (R6),#0                 ;VERIFY CONTENTS OF STAACK
3794  014016  001003                          BNE     12$                     ;BRANCH IF DATA ON STACK INCORRECT
3795  014020  022704  013622                  CMP     #MJSRA-2,R4             ;SEE IF CORRECT RETURN ADDRESS
3796  014024  001401                          BEQ     MJSR3B                  ;BRANCH IF GOOD
3797  014026  104001                  12$:    ERROR   ;1              ;CPU ERROR
3798                                                                  ;JSR MODE 3 FAILED
3799  014030  005237  003032          MJSR3B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3800  014034  013706  003034                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3801  014040  012704  052525                  MOV     #052525,R4              ;SETUP R4 DATA
3802  014044  012701  013710                  MOV     #MJSR4+2,R1             ;SETUP JSR VECTOR
3803  014050  000257                          CCC                             ;CLEAR CONDITION CODES
3804  014052  004441                          JSR     R4,-(R1)                        ;JSR MODE 4
3805                                   ;
3806                                   ;
3807  014054  022737  000006  003032  MJSR6:  CMP     #6,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3808  014062  001401                          BEQ     MJSR6A                  ;BRANCH IF GOOD
3809  014064  104001                  13$:    ERROR   ;1              ;CPU ERROR
3810                                                                  ;MODE 6 JUMPED TO OUT OF SEQUENCE
3811  014066  023706  003036          MJSR6A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3812  014072  001006                          BNE     14$                     ;BRANCH IF STACK INCORRECT
3813  014074  021627  123456                  CMP     (R6),#123456            ;VERIFY CONTENTS OF STACK
3814  014100  001003                          BNE     14$                     ;BRANCH IF DATA ON STACK INCORRECT
3815  014102  022704  014222                  CMP     #MJSR7,R4               ;SEE IF CORRECT RETURN ADDRESS
3816  014106  001401                          BEQ     MJSR6B                  ;BRANCH IF GOOD
3817  014110  104001                  14$:    ERROR   ;1              ;CPU ERROR
3818                                                                  ;JSR MODE 6 FAILED
3819  014112  005237  003032          MJSR6B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3820  014116  013706  003034                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3821  014122  012704  177773                  MOV     #-5,R4                  ;SETUP R4 DATA
3822  014126  012701  014146                  MOV     #MJSRC+10,R1            ;SETUP JSR VECTOR
3823  014132  004471  177770                  JSR     R4,@-10(R1)             ;JSR MODE 7
```

```
3824  014136  014222                  MJSRC:  .WORD   MJSR7                   ;JSR VECTOR
3825                                  ;
3826                                  ;
3827  014140  022737  000005  003032  MJSR5:  CMP     #5,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3828  014146  001401                          BEQ     MJSR5A                  ;BRANCH IF GOOD
3829  014150  104001                  15$:    ERROR   .1              ;CPU ERROR
3830                                                                  ;MODE 5 JUMPED TO OUT OF SEQUENCE
3831  014152  023706  003036          MJSR5A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3832  014156  001006                          BNE     16$                     ;BRANCH IF STACK INCORRECT
3833  014160  021627  000377                  CMP     (R6),#377               ;VERIFY CONTENTS OF STACK
3834  014164  001003                          BNE     16$                     ;BRANCH IF DATA ON STACK INCORRECT
3835  014166  022704  013766                  CMP     #MJSR8-2,R4             ;SEE IF CORRECT RETURN ADDRESS
3836  014172  001401                          BEQ     MJSR5B                  ;BRANCH IF GOOD
3837  014174  104001                  16$:    ERROR   .1              ;CPU ERROR
3838                                                                  ;JSR MODE 5 FAILED
3839  014176  005237  003032          MJSR5B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3840  014202  013706  003034                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3841  014206  012704  123456                  MOV     #123456,R4              ;SETUP DATA IN R4
3842  014212  012701  014064                  MOV     #MJSR6+10,R1            ;SETUP JSR VECTOR
3843  014216  004461  177770                  JSR     R4,-10(R1)              ;JUMP MODE 6
3844                                  ;
3845                                  ;
3846  014222  022737  000007  003032  MJSR7:  CMP     #7,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3847  014230  001401                          BEQ     MJSR7A                  ;BRANCH IF GOOD
3848  014232  104001                  17$:    ERROR   .1              ;CPU ERROR
3849                                                                  ;MODE 7 JUMPED TO OUT OF SEQUENCE
3850  014234  023706  003036          MJSR7A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3851  014240  001006                          BNE     18$                     ;BRANCH IF STACK INCORRECT
3852  014242  021627  177773                  CMP     (R6),#-5                ;VERIFY CONTENTS OF STAACK
3853  014246  001003                          BNE     18$                     ;BRANCH IF DATA ON STACK INCORRECT
3854  014250  022704  014136                  CMP     #MJSR5-2,R4             ;SEE IF CORRECT RETURN ADDRESS
3855  014254  001401                          BEQ     MJSR7E                  ;BRANCH IF GOOD
3856  014256  104001                  18$:    ERROR   .1              ;CPU ERROR
3857                                                                  ;JSR MODE 7 FAILED
3858  014260                          MJSR7E:
3859  014260  013706  003034                  MOV     @#SPS,R6                ;REPLACE STACK
3860
3861                                  ;
3862  014264                          MJRA:
3863
3864                                  ;       TEST JSR MODES 27, 37, 67, 77
3865  014264  012737  000001  003032          MOV     #1,@#SEQ                ;SETUP SEQUENCER
3866  014272  010637  003034                  MOV     R6,@#SPS                ;SAVE STACK ADDRESS
3867  014276  010637  003036                  MOV     R6,@#SPSJ               ;SAVE STACK DECRIMENT ADDRESS
3868  014302  162737  000002  003036          SUB     #2,@#SPSJ               ;
3869  014310  012704  177777                  MOV     #-1,R4                  ;SETUP R4 DATA
3870  014314  004427  000240                  JSR     R4,#240                 ;EXECUTE A JSR MODE 27
3871                                  ;
3872  014320  022737  000001  003032  MJR27:  CMP     #1,@#SEQ                ;VERIFY COERRECT TEST SEQUENCE
3873  014326  001011                          BNE     1$                      ;INCORRECT TEST SEQUENCE
3874  014330  023706  003036                  CMP     @#SPSJ,R6               ;VERIFY STACK POINTER
3875  014334  001006                          BNE     1$
3876  014336  021627  177777                  CMP     (R6),#-1                ;VERIFY R4 GOT LOADED ON THE STACK
3877  014342  001003                          BNE     1$                      ;BRANCH IF INCORRECT STACK CONTENTS
3878  014344  020427  014320                  CMP     R4,#MJR27               ;VERIFY CORRECT RETURN ADDRESS
3879  014350  001401                          BEQ     MJR27A                  ;BRANCH IF GOOD RETURN ADDRESS ON STACK
```

```
3880  014352  104001                  1$:     ERROR   ;1              ;CPU ERROR
3881                                                                  ;MODE 27 FAILED
3882  014354  005237  003032          MJR27A: INC     @#SEQ                   ;UPDATE SEQUENCER
3883  014360  012704  152525                  MOV     #152525,R4              ;SETUP R4 TEST DATA
3884  014364  013706  003034                  MOV     @#SPS,R6                ;RESET STACK POINTER
3885  014370  004437  014456                  JSR     R4,@#MJR37              ;JSR MODE 37
3886  014374  000000                  MJR27B: HALT
3887                                  ;
3888  014376  023727  003032  000003  MJR67:  CMP     @#SEQ,#3                ;VERIFY TEST SEQUENCE
3889  014404  001011                          BNE     2$                      ;INCORRECT TEST SEQUENCE
3890  014406  023706  003036                  CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3891  014412  001006                          BNE     2$                      ;INCORRECT STACK DECRIMENT
3892  014414  021627  000125                  CMP     (R6),#125               ;VERFIFY STACK WAS LOADED
3893  014420  001003                          BNE     2$
3894  014422  020427  014532                  CMP     R4,#MJR67               ;VERIFY RETURN ADDRESS
3895  014426  001401                          BEQ     MJR67A                  ;BRANCH IF GOOD
3896  014430  104001                  2$:     ERROR   ;1              ;CPU ERROR
3897                                                                  ;MODE 67 FAILED
3898  014432  005237  003032          MJR67A: INC     @#SEQ                   ;UPDATE SEQUENCER
3899  014436  013706  003034                  MOV     @#SPS,R6                ;RESET STACK
3900  014442  012704  000001                  MOV     #1,R4                   ;SETUP R4 DATA
3901  014446  004477  000002                  JSR     R4,@MJR6B               ;JSR MODE 77
3902  014452  000000                  MJR6A:  HALT
3903  014454  014532                  MJR6B:  .WORD   MJR77                   ;DATA FOR MODE 77 JUMP
3904                                  ;
3905  014456  023727  003032  000002  MJR37:  CMP     @#SEQ,#2                ;VERIFY TEST SEQUENCE
3906  014464  001011                          BNE     2$                      ;INCORRECT TEST SEQUENCE
3907  014466  023706  003036                  CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3908  014472  001006                          BNE     2$                      ;INCORRECT STACK DECRIMENT
3909  014474  021627  152525                  CMP     (R6),#152525            ;VERFIFY STACK WAS LOADED
3910  014500  001003                          BNE     2$
3911  014502  020427  014374                  CMP     R4,#MJR27B              ;VERIFY RETURN ADDRESS
3912  014506  001401                          BEQ     MJR37A                  ;BRANCH IF GOOD
3913  014510  104001                  2$:     ERROR   ;1              ;CPU ERROR
3914                                                                  ;MODE 37 FAILED
3915  014512  005237  003032          MJR37A: INC     @#SEQ                   ;UPDATE SEQUENCER
3916  014516  013706  003034                  MOV     @#SPS,R6                ;RELOAD STACK
3917  014522  012704  000125                  MOV     #125,R4                 ;SETUP R4 TEST DATA
3918  014526  004467  177644                  JSR     R4,MJR67                ;JSR MODE 6
3919                                  ;
3920  014532  023727  003032  000004  MJR77:  CMP     @#SEQ,#4                ;VERIFY TEST SEQUENCE
3921  014540  001011                          BNE     2$                      ;INCORRECT TEST SEQUENCE
3922  014542  023706  003036                  CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3923  014546  001006                          BNE     2$                      ;INCORRECT STACK DECRIMENT
3924  014550  021627  000001                  CMP     (R6),#1                 ;VERFIFY STACK WAS LOADED
3925  014554  001003                          BNE     2$
3926  014556  020427  014452                  CMP     R4,#MJR6A               ;VERIFY RETURN ADDRESS
3927  014562  001401                          BEQ     MJR77A                  ;BRANCH IF GOOD
3928  014564  104001                  2$:     ERROR   ;1              ;CPU ERROR
3929                                                                  ;MODE 77 FAILED
3930  014566                          MJR77A:
3931                                  ;
3932  014566  013706  003034                  MOV     @#SPS,R6                ;RESET STACK
3933
3934                                  ;
3935  014572                          MRTS:
```

J6


```
3936
3937                                    ;       TEST RTS AND RTS R6
3938  014572  012706  001000                    MOV     #STBOT,R6               ;INSURE VALID STACK
3939  014576  012746  123456                    MOV     #123456,-(R6)           ;SETUP TEST REGISTER
3940  014602  012703  014612                    MOV     #RTS1,R3                ;SETUP TEST PC
3941  014606  000203                            RTS     R3                      ;**TEST INSTRUCTION
3942  014610  104001                            ERROR   .1              ;CPU ERROR
3943                                                                    ;INCORRECT PC ON RTS
3944  014612  022703  123456          RTS1:     CMP     #123456,R3
3945  014616  001401                            BEQ     RTS6                    ;BRANCH IF GOOD
3946  014620  104001                            ERROR   .1              ;CPU ERROR
3947                                                                    ;REGISTER CONTENTS INCORRECT
3948                                            ;
3949                                            ;THIS TEST CHECKS AN UN-TESTED PLA TERM
3950                                            ;
3951  014622  010601                    RTS6:   MOV     R6,R1                   ;SAVE STACK IN R1
3952  014624  012705  014636                    MOV     #1$,R5                  ;MOVE EXPECTED RETURN ADDR TO R5
3953  014630  010506                            MOV     R5,R6                   ;MOVE RETURN ADDR TO R6
3954  014632  000206                            RTS     R6                      ;
3955  014634  104001                            ERROR   .1              ;CPU ERROR
3956                                                                    ;ERROR! RTS NOT EXECUTED
3957  014636  021506                    1$:     CMP     (R5),R6                 ;IS R6=31506?
3958  014640  001401                            BEQ     RTSE                    ;IF IT IS THEN GO TO END OF TEST
3959  014642  104001                            ERROR   .1              ;CPU ERROR
3960                                                                    ;ERROR! WRONG ADDR IN R6
3961  014644  010106                    RTSE:   MOV     R1,R6                   ;RESTORE STACK
3962
3963  014646                            TSMMUO:
3964                                    ;       SETUP AND TEST KERNEL, SUPERVISOR AND USER STACKS
3965  014646  012737  040000  177776            MOV     #40000,@#177776         ;SET PS TO SUP MODE
3966  014654  012706  177777                    MOV     #177777,R6              ;INIT SUP STACK TO ALL ONES
3967  014660  022706  177777                    CMP     #177777,R6              ;ARE ALL BITS SET
3968  014664  001401                            BEQ     1$                      ;YES GO ON
3969                                                                    ;NO GO TO ERROR
3970  014666  104001                            ERROR   .1              ;CPU ERROR
3971  014670  005006                    1$:     CLR     R6                      ;SET SUP STACK TO ALL ZEROES
3972  014672  022706  000000                    CMP     #0,R6                   ;ARE ALL BITS CLEARED
3973  014676  001401                            BEQ     2$                      ;YES GO ON
3974                                                                    ;NO GO TO ERROR
3975  014700  104001                            ERROR   .1              ;CPU ERROR
3976  014702  012706  125252            2$:     MOV     #125252,R6              ;SET SUP STACK TO ALTERNATING PATTERN
3977  014706  022706  125252                    CMP     #125252,R6              ;IS SUP SP CORRECT
3978  014712  001401                            BEQ     3$                      ;YES GO ON
3979                                                                    ;NO GO TO ERROR
3980  014714  104001                            ERROR   .1              ;CPU ERROR
3981  014716  012706  052525            3$:     MOV     #52525,R6               ;SET SUP STACK TO ALTERNATING PATTERN
3982  014722  022706  052525                    CMP     #52525,R6               ;IS SUP SP CORRECT
3983  014726  001401                            BEQ     4$                      ;YES GO ON
3984                                                                    ;NO GO TO ERROR
3985  014730  104001                            ERROR   .1              ;CPU ERROR
3986  014732  012706  000700            4$:     MOV     #700,R6                 ;SETUP SUP SP
3987  014736  012737  140000  177776            MOV     #140000,@#177776        ;SET PS TO USER MODE
3988  014744  012706  177777                    MOV     #177777,R6              ;INIT USER STACK TO ALL ONES
3989  014750  022706  177777                    CMP     #177777,R6              ;ARE ALL BITS SET
3990  014754  001401                            BEQ     5$                      ;YES GO ON
3991                                                                    ;NO GO TO ERROR
```

```
3992  014756  104001                          ERROR   ;1              ;CPU ERROR
3993  014760  005006                  5$:     CLR     R6                      ;SET USER STACK TO ALL ZEROES
3994  014762  022706  000000                  CMP     #0,R6                   ;ARE ALL BITS CLEARED
3995  014766  001401                          BEQ     6$                      ;YES GO ON
3996                                                                          ;NO GO TO ERROR
3997  014770  104001                          ERROR   ;1              ;CPU ERROR
3998  014772  012706  125252          6$:     MOV     #125252,R6              ;SET USER STACK TO ALTERNATING PATTERN
3999  014776  022706  125252                  CMP     #125252,R6              ;IS USER SP CORRECT
4000  015002  001401                          BEQ     7$                      ;YES GO ON
4001                                                                          ;NO GO TO ERROR
4002  015004  104001                          ERROR   ;1              ;CPU ERROR
4003  015006  012706  052525          7$:     MOV     #52525,R6               ;SET USER STACK TO ALTERNATING PATTERN
4004  015012  022706  052525                  CMP     #52525,R6               ;IS USER SP CORRECT
4005  015016  001401                          BEQ     8$                      ;YES GO ON
4006                                                                          ;NO GO TO ERROR
4007  015023  104001                          ERROR   ;1              ;CPU ERROR
4008  015022  012706  000600          8$:     MOV     #600,R6                 ;SETUP USER SP
4009  015026  005037  177776                  CLR     @#177776                ;SET PS TO KER MODE
4010  015032  012706  001000                  MOV     #STBOT,R6               ;SETUP KERNEL SP
4011  015036  005037  000700                  CLR     @#700                   ;CLEAR FIRST WORDS OF SUP, KER, AND USE STACKS
4012  015042  005037  000600                  CLR     @#600                   ;
4013  015046  005037  001000                  CLR     @#STBOT                 ;
4014  015052  004767  000054                  JSR     PC,CHECK                ; TEST KER, SUP, AND USE STACKS
4015  015056  012737  040000  177776  RET1:   MOV     #40000,@#177776         ;SET PSW TO SUP MODE
4016  015064  022706  000700                  CMP     #700,R6                 ;IS SUP SP CORRECT
4017  015070  001401                          BEQ     1$                      ;YES GO ON
4018                                                                          ;NO GO TO ERROR
4019  015072  104001                          ERROR   ;1              ;CPU ERROR
4020  015074  012737  140000  177776  1$:     MOV     #140000,@#177776        ;SET PSW TO USE MODE
4021  015102  022706  000600                  CMP     #600,R6                 ;IS USE SP CORRECT
4022  015106  001401                          BEQ     2$                      ;YES GO ON
4023                                                                          ;NO GO TO ERROR
4024  015110  104001                          ERROR   ;1              ;CPU ERROR
4025  015112  005037  177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
4026  015116  022706  001000                  CMP     #STBOT,R6               ;IS KER SP CORRECT
4027  015122  001401                          BEQ     3$                      ;YES GO ON
4028                                                                          ;NO GO TO ERROR
4029  015124  104001                          ERROR   ;1              ;CPU ERROR
4030  015126                          3$:
4031  015126  000167  000206                  JMP     MTS0
4032                                          ;
4033                                          ;ROUTINE TO CHECK STACKS AFTER TWO RTS STATEMENTS
4034                                          ;
4035  015132  012737  040000  177776  CHECK:  MOV     #40000,@#177776         ;SET PSW TO SUP MODE
4036  015140  004767  000044                  JSR     PC,CHECK1               ; TEST SUP, KER, AND USE STACKS
4037  015144  022716  000000          RET2:   CMP     #0,(SP)                 ;IS SUP STACK CLEARED
4038  015150  001401                          BEQ     1$                      ;YES GO ON
4039                                                                          ;NO GO TO ERROR
4040  015152  104001                          ERROR   ;1              ;CPU ERROR
4041  015154  012737  140000  177776  1$:     MOV     #140000,@#177776        ;SET PSW TO USE MODE
4042  015162  022716  000000                  CMP     #0,(SP)                 ;IS USE STACK CLEARED
4043  015166  001401                          BEQ     2$                      ;YES GO ON
4044                                                                          ;NO GO TO ERROR
4045  015170  104001                          ERROR   ;1              ;CPU ERROR
4046  015172  005037  177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
4047  015176  022716  015056                  CMP     #RET1,(SP)              ;DOES KER STACK HAVE CORRECT DATA
```

```
4048  015202  001401                          BEQ     3$                      ;YES GO ON
4049                                                                          ;NO GO TO ERROR
4050  015204  104001                          ERROR   +1              ;CPU ERROR
4051  015206  000207                  3$:     RTS     PC                      ;RETURN
4052                                          ;
4053                                          ;ROUTINE TO CHECK STACKS AFTER ONE RTS
4054                                          ;
4055  015210  012737  140000  177776  CHECK1: MOV     #140000,@#177776        ;SET PSW TO USE MODE
4056  015216  004767  000044                  JSR     PC,CHECK2               ; TEST KER, SUP, AND USE STACKS
4057  015222  022716  000000          RET3:   CMP     #0,(SP)                 ;IS USE STACK CLEARED
4058  015226  001401                          BEQ     1$                      ;YES GO ON
4059                                                                          ;NO GO TO ERROR
4060  015230  104001                          ERROR   +1              ;CPU ERROR
4061  015232  005037  177776          1$:     CLR     @#177776                ;SET PSW TO KER MODE
4062  015236  022716  015056                  CMP     #RET1,(SP)              ;IS KER STACK CORRECT
4063  015242  001401                          BEQ     2$                      ;YES GO ON
4064                                                                          ;NO GO TO ERROR
4065  015244  104001                          ERROR   +1              ;CPU ERROR
4066  015246  012737  040000  177776  2$:     MOV     #40000,@#177776         ;SET PSW TO SUP MODE
4067  015254  022716  015144                  CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
4068  015260  001401                          BEQ     3$                      ;YES GO ON
4069                                                                          ;NO GO TO ERROR
4070  015262  104001                          ERROR   +1              ;CPU ERROR
4071  015264  000207                  3$:     RTS     PC                      ;RETURN
4072                                          ;
4073                                          ;ROUTINE TO CHECK STACKS AFTER ZERO RTS
4074                                          ;
4075  015266  022716  015222          CHECK2: CMP     #RET3,(SP)              ;IS USE STACK CORRECT
4076  015272  001401                          BEQ     1$                      ;YES GO ON
4077                                                                          ;NO GO TO ERROR
4078  015274  104001                          ERROR   +1              ;CPU ERROR
4079  015276  012737  040000  177776  1$:     MOV     #40000,@#177776         ;SET PSW TO SUP MODE
4080  015304  022716  015144                  CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
4081  015310  001401                          BEQ     2$                      ;YES GO ON
4082                                                                          ;NO GO TO ERROR
4083  015312  104001                          ERROR   +1              ;CPU ERROR
4084  015314  005037  177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
4085  015320  022716  015056                  CMP     #RET1,(SP)              ;IS KER STACK CORRECT
4086  015324  001401                          BEQ     3$                      ;YES GO ON
4087                                                                          ;NO GO TO ERROR
4088  015326  104001                          ERROR   +1              ;CPU ERROR
4089  015330  012737  140000  177776  3$:     MOV     #140000,@#177776        ;SET PSW TO USE MODE
4090  015336  000207                          RTS     PC                      ;RETURN
4091                                          ;
4092  015340                          MTSO:
4093  015340                          MMVCC:
4094
4095                                  ;       TEST MOV CONDITION CODES - **0-
4096  015340  000277                          SCC                             ;
4097  015342  000244                          CLZ                             ;CC=1011
4098  015344  012704  000000                  MOV     #0,R4                   ;CC=0101, R4=0
4099  015350  100403                          BMI     1$                      ;ERROR IF N FLAG
4100  015352  102402                          BVS     1$                      ;ERROR IF V FLAG SET
4101  015354  103001                          BCC     1$                      ;ERROR IF C FLAG CLEAR
4102  015356  001401                          BEQ     2$                      ;SKIP IF Z FLAG SET
4103                                                                          ;CC SHOULD=0101
```

```
                        ************ DOUBLE OPERAND TESTS ****************                SEQ 0077

 4104  015360  104001                1$:     ERROR   ;1              ;CPU ERROR
 4105  015362  000277                2$:     SCC
 4106  015364  000251                        .WORD   251                     ;CC=0110
 4107  015366  012704  100000                MOV     #100000,R4              ;R4=100000, CC=1000
 4108  015372  001403                        BEQ     3$                      ;ERROR IF Z SET
 4109  015374  102402                        BVS     3$                      ;ERROR IF V SET
 4110  015376  103401                        BCS     3$                      ;ERROR IF C SET
 4111  015400  100401                        BMI     4$                      ;EXIT IF N SET
 4112  015402  104001                3$:     ERROR   ;1              ;CPU ERROR
 4113                                                                  ;CC SHOULD= 1000
 4114  015404                        4$:
 4115
 4116
 4117  015404                        MBTCC:
 4118
 4119                                ;       TEST BIT CONDITION CODES - **0-
 4120  015404  005004                        CLR     R4
 4121  015406  005104                        COM     R4                      ;R4=-1
 4122  015410  000277                        SCC                             ;
 4123  015412  000244                        CLZ                             ;CC=1011
 4124  015414  032704  000000                BIT     #0,R4                   ;CC=0101
 4125  015420  100403                        BMI     1$                      ;ERROR IF N FLAG
 4126  015422  102402                        BVS     1$                      ;ERROR IF V FLAG SET
 4127  015424  103001                        BCC     1$                      ;ERROR IF C FLAG CLEAR
 4128  015426  001401                        BEQ     2$                      ;SKIP IF Z FLAG SET
 4129  015430  104001                1$:     ERROR   ;1              ;CPU ERROR
 4130                                                                  ;CC SHOULD=0101
 4131  015432  000277                2$:     SCC
 4132  015434  000251                        .WORD   251                     ;CC=0110
 4133  015436  032704  100000                BIT     #100000,R4              ;CC=1000
 4134  015442  001403                        BEQ     3$                      ;ERROR IF Z SET
 4135  015444  102402                        BVS     3$                      ;ERROR IF V SET
 4136  015446  103401                        BCS     3$                      ;ERROR IF C SET
 4137  015450  100401                        BMI     4$                      ;EXIT IF N SET
 4138  015452  104001                3$:     ERROR   ;1              ;CPU ERROR
 4139                                                                  ;CC SHOULD= 1000
 4140  015454                        4$:
 4141
 4142
 4143  015454                        MBCCC:
 4144
 4145                                ;       TEST BIC CONDITION CODES - **0-
 4146  015454  005004                        CLR     R4
 4147  015456  005104                        COM     R4                      ;R4=-1
 4148  015460  000277                        SCC                             ;
 4149  015462  000244                        CLZ                             ;CC=1011
 4150  015464  042704  177777                BIC     #177777,R4              ;CC=0101
 4151  015470  100403                        BMI     1$                      ;ERROR IF N FLAG
 4152  015472  102402                        BVS     1$                      ;ERROR IF V FLAG SET
 4153  015474  103001                        BCC     1$                      ;ERROR IF C FLAG CLEAR
 4154  015476  001401                        BEQ     2$                      ;SKIP IF Z FLAG SET
 4155  015500  104001                1$:     ERROR   ;1              ;CPU ERROR
 4156                                                                  ;CC SHOULD=0101
 4157  015502  005104                2$:     COM     R4                      ;R4=-1
 4158  015504  000277                        SCC
 4159  015506  000251                        .WORD   251                     ;CC=0110
```

```
 4160  015510  042704  077777                BIC     #77777,R4               ;CC=1000
 4161  015514  001403                        BEQ     3$                      ;ERROR IF Z SET
 4162  015516  102402                        BVS     3$                      ;ERROR IF V SET
 4163  015520  103401                        BCS     3$                      ;ERROR IF C SET
 4164  015522  100401                        BMI     4$                      ;EXIT IF N SET
 4165  015524  104001              3$:       ERROR   +1              ;CPU ERROR
 4166                                                                ;CC SHOULD= 1000
 4167  015526                    4$:
 4168
 4169
 4170  015526                    MBSCC:
 4171
 4172                            ;       TEST BIS CONDITION CODES
 4173  015526  005004                        CLR     R4                      ;R4=0
 4174  015530  000277                        SCC
 4175  015532  000246                        .WORD   246                     ;CC=1001
 4176  015534  052704  000000                BIS     #0,R4                   ;R4=0, CC=0101
 4177  015540  100403                        BMI     1$                      ;ERROR IF MINUS
 4178  015542  102402                        BVS     1$                      ;ERROR IF V SET
 4179  015544  103001                        BCC     1$                      ;ERROR IF C CLEAR
 4180  015546  001401                        BEQ     2$                      ;BRANCH IF GOOD
 4181  015550  104001              1$:       ERROR   +1              ;CPU ERROR
 4182                                                                ;BIS CC FAILED
 4183  015552  000277              2$:       SCC
 4184  015554  000241                        CLC                             ;CC=1110
 4185  015556  052704  100076                BIS     #100076,R4              ;R4=100076, CC=1000
 4186  015562  001403                        BEQ     3$                      ;ERROR IF Z SET
 4187  015564  102402                        BVS     3$                      ;ERROR IF V SET
 4188  015566  103401                        BCS     3$                      ;ERROR IF C SET
 4189  015570  100401                        BMI     4$                      ;BRANCH IF GOOD
 4190  015572  104001              3$:       ERROR   +1              ;CPU ERROR
 4191                                                                ;BAD BIS CC
 4192  015574                    4$:
 4193
 4194
 4195  015574                    MDCCC:
 4196
 4197                            ;       TEST DEC, INC CONDITION CODES
 4198  015574  012704  077777                MOV     #77777,R4               ;R4=77777
 4199  015600  000257                        CCC
 4200  015602  000261                        SEC                             ;CC=0001
 4201  015604  005204                        INC     R4                      ;R4=100000, CC=0011
 4202  015606  001403                        BEQ     1$                      ;ERROR IF ZERO
 4203  015610  100002                        BPL     1$                      ;ERROR IF POSITIVE
 4204  015612  102001                        BVC     1$                      ;ERROR IF V CLEAR
 4205  015614  103401                        BCS     2$                      ;BRANCH IF GOOD
 4206  015616  104001              1$:       ERROR   +1              ;CPU ERROR
 4207                                                                ;INC FAILED
 4208  015620  000257              2$:       CCC
 4209  015622  005204                        INC     R4                      ;R4=100001, CC=1000
 4210  015624  103413                        BCS     3$                      ;ERROR IF C SET
 4211  015626  102412                        BVS     3$                      ;ERROR IF V SET
 4212  015630  005304                        DEC     R4                      ;R4=100000, CC=1000
 4213  015632  102410                        BVS     3$                      ;ERROR IF V SET
 4214  015634  103407                        BCS     3$                      ;ERROR IF C SET
 4215  015636  000277                        SCC                             ;
```

```
 4216  015640  000252                          .WORD   252                     ;CC=0101
 4217  015642  005304                          DEC     R4                      ;R4=77777, CC=1011
 4218  015644  001403                          BEQ     3$                      ;ERROR IF Z SET
 4219  015646  102002                          BVC     3$                      ;ERROR IF V CLEAR
 4220  015650  103001                          BCC     3$                      ;ERROR IF C CLEAR
 4221  015652  100001                          BPL     4$                      ;BRANCH IF GOOD
 4222  015654  104001                  3$:     ERROR   #1              ;CPU ERROR
 4223                                                                  ;BAD CC
 4224  015656                          4$:
 4225
 4226
 4227  015656                          MCTSCC:
 4228
 4229                                  ;       TEST CLR, TST, SWAB CONDITION CODES
 4230                                  ;;**************************************************************
 4231                                  ;0100 - **00 - **00
 4232  015656  000277                          SCC
 4233  015660  000244                          CLZ                             ;CC=1011
 4234  015662  005004                          CLR     R4                      ;R4=0, CC=0100
 4235  015664  100403                          BMI     1$                      ;ERROR IF MINUS
 4236  015666  102402                          BVS     1$                              ;ERROR IF V SET
 4237  015670  103401                          BCS     1$                      ;ERROR IF C SET
 4238  015672  001401                          BEQ     2$                      ;BRANCH IF GOOD
 4239  015674  104001                  1$:     ERROR   #1              ;CPU ERROR
 4240                                                                  ;BAD CC ON CLR
 4241  015676  005104                  2$:     COM     R4                      ;R4=-1
 4242  015700  000277                          SCC                             ;CC=1111
 4243  015702  005704                          TST     R4                      ;CC=1000
 4244  015704  001403                          BEQ     3$                      ;ERROR IF Z SET
 4245  015706  102402                          BVS     3$                      ;ERROR IF V SET
 4246  015710  103401                          BCS     3$                      ;ERROR IF C SET
 4247  015712  100401                          BMI     4$                      ;BRANCH IF GOOD
 4248  015714  104001                  3$:     ERROR   #1              ;CPU ERROR
 4249                                                                  ;BAD TST CC
 4250  015716  000277                  4$:     SCC                             ;CC=1111
 4251  015720  000304                          SWAB    R4                      ;CC=1000
 4252  015722  102402                          BVS     5$                      ;ERROR IF V SET
 4253  015724  103401                          BCS     5$                      ;ERROR IF C SET
 4254  015726  100401                          BMI     6$                      ;BRANCH IF GOOD
 4255  015730  104001                  5$:     ERROR   #1              ;CPU ERROR
 4256                                                                  ;BAD SWAB CC
 4257  015732                          6$:
 4258
 4259
 4260  015732                          MADCC:
 4261
 4262                                  ;       TEST ADD CONDITION CODES
 4263  015732  012704  077777                  MOV     #77777,R4               ;R4=77777
 4264  015736  012701  000001                  MOV     #1,R1                   ;R1=1
 4265  015742  000257                          CCC                             ;CC=0000
 4266  015744  060401                          ADD     R4,R1                   ;77777 + 1 = 100000 IN R1
 4267  015746  102003                          BVC     1$                      ;ERROR IF V CLEAR
 4268  015750  103402                          BCS     1$                      ;ERROR IF CARRY
 4269  015752  001401                          BEQ     1$                      ;ERROR IF Z SET
 4270  015754  100401                          BMI     2$                      ;BRANCH IF GOOD
 4271  015756  104001                  1$:     ERROR   #1              ;CPU ERROR
```

```
4272
4273  015760  005204                  2$:     INC     R4                      ;CC SHOULD =1010
4274  015762  060401                          ADD     R4,R1                   ;R4=100000
4275  015764  102002                          BVC     3$                      ;100000 + 100000 = 0 IN R1
4276  015766  103001                          BCC     3$                      ;ERROR IF V CLEAR
4277  015770  001401                          BEQ     4$                      ;ERROR IF CARRY  CLEAR
4278  015772  104001                  3$:     ERROR   +1                      ;BRANCH IF GOOD
4279                                                                          ;CPU ERROR
4280  015774  060401                  4$:     ADD     R4,R1                   ;CC SHOULD = 0111
4281  015776  102402                          BVS     5$                      ;0 + 100000 = 100000
4282  016000  103401                          BCS     5$                      ;ERROR IF V SET
4283  016002  100401                          BMI     6$                      ;ERROR IF  SET
4284  016004  104001                  5$:     ERROR   +1                      ;BRANCH IF GOOD
4285                                                                          ;CPU ERROR
4286  016006                          6$:                                     ;CC SHOULD = 1000
4287
4288
4289  016006                          MACCC:
4290
4291                                  ;       TEST ADC CONDITION CODES
4292  016006  012704  177777                 MOV     #177777,R4              ;R4=177777
4293  016012  000277                          SCC                             ;CC=1111
4294  016014  005504                          ADC     R4                      ;R4=0 CC=0101
4295  016016  100403                          BMI     1$                      ;ERROR IF MINUS
4296  016020  102402                          BVS     1$                      ;ERROR IF V SET
4297  016022  103001                          BCC     1$                      ;ERROR IF C SET
4298  016024  001401                          BEQ     2$                      ;BRANCH IF GOOD
4299  016026  104001                  1$:     ERROR   +1                      ;CPU ERROR
4300                                                                          ;BAD ADC
4301  016030  012704  077777          2$:     MOV     #077777,R4              ;R4=077777
4302  016034  000277                          SCC                             ;
4303  016036  000242                          CLV                             ;CC=1101
4304  016040  005504                          ADC     R4                      ;R4=100000 CC=1010
4305  016042  100003                          BPL     3$                      ;ERROR IF PLUS
4306  016044  103402                          BCS     3$                      ;ERROR IF C SET
4307  016046  001401                          BEQ     3$                      ;ERROR IF ZERO
4308  016050  102401                          BVS     4$                      ;BRANCH IF GOOD
4309  016052  104001                  3$:     ERROR   +1                      ;CPU ERROR
4310                                                                          ;BAD ADC
4311  016054  000277                  4$:     SCC                             ;CC=1111
4312  016056  005504                          ADC     R4                      ;R4=100000 CC=1000
4313  016060  102402                          BVS     5$                      ;ERROR IF V SET
4314  016062  103401                          BCS     5$                      ;ERROR IF C SET
4315  016064  100401                          BMI     6$                      ;BRANCH IF GOOD
4316  016066  104001                  5$:     ERROR   +1                      ;CPU ERROR
4317                                                                          ;BAD ADC CC SHOULD= 1000
4318  016070                          6$:
4319
4320
4321  016070                          MNCCCC:
4322
4323                                  ;       TEST NEG, CMP, COM CONDITION CODES
4324  016070  012704  077777                 MOV     #077777,R4              ;R4=77777
4325  016074  000257                          CCC                             ;CC=0000
4326  016076  005404                          NEG     R4                      ;R4=100001      CC=1001
4327  016100  102403                          BVS     1$                      ;ERROR IF V SET
```

```
4328  016102  103002                  BCC     1$                      ;ERROR IF C CLEAR
4329  016104  001401                  BEQ     1$                      ;ERROR IF Z SET
4330  016106  100401                  BMI     2$                      ;BRANCH IF GOOD
4331  016110  104001          1$:     ERROR   ·1              ;CPU ERROR
4332                                                          ;BAD NEGATE
4333  016112  005004          2$:     CLR     R4                      ;R4=0
4334  016114  000257                  CCC                             ;CC=0000
4335  016116  005404                  NEG     R4                      ;CC=0101
4336  016120  100403                  BMI     3$                      ;ERROR IF N SET
4337  016122  103402                  BCS     3$                      ;ERROR IF C SET
4338  016124  102401                  BVS     3$                      ;ERROR IF V SET
4339  016126  001401                  BEQ     4$                      ;BRANCH IF GOOD
4340  016130  104001          3$:     ERROR   ·1              ;CPU ERROR
4341                                                          ;BAD NEG
4342  016132  012704  077777  4$:     MOV     #77777,R4               ;R4=77777
4343  016136  012701  170000          MOV     #170000,R1              ;R1=170000
4344  016142  000257                  CCC                             ;CC=0000
4345  016144  020401                  CMP     R4,R1                   ;77777 - 170000 = 107777 CC= 1011
4346  016146  102003                  BVC     5$                      ;ERROR IF V CLEAR
4347  016150  103002                  BCC     5$                      ;ERROR IF C CLEAR
4348  016152  001401                  BEQ     5$                      ;ERROR IF ZERO
4349  016154  100401                  BMI     6$                      ;BRANCH IF GOOD
4350  016156  104001          5$:     ERROR   ·1              ;CPU ERROR
4351                                                          ;BAD CMP
4352  016160  000257          6$:     CCC
4353  016162  005101                  COM     R1                      ;R1=7777
4354  016164  100403                  BMI     7$                      ;ERROR IF MINUS
4355  016166  001402                  BEQ     7$                      ;ERROR IF ZERO
4356  016170  103001                  BCC     7$                      ;ERROR IF CARRY
4357  016172  102001                  BVC     8$                      ;BRANCH IF GOOD
4358  016174  104001          7$:     ERROR   ·1              ;CPU ERROR
4359                                                          ;BAD COM
4360  016176  000277          8$:     SCC
4361  016200  005101                  COM     R1
4362  016202  100401                  BMI     10$                     ;BRANCH IF GOOD
4363  016204  104001          9$:     ERROR   ·1              ;CPU ERROR
4364                                                          ;BAD COM
4365  016206                  10$:
4366
4367
4368  016206                  MSBCC:
4369
4370                          ;       TEST SUB CONDITION CODES
4371  016206  012704  077775          MOV     #77775,R4               ;R4=77775
4372  016212  000257                  CCC                             ;CC=0000
4373  016214  162704  137757          SUB     #137757,R4              ;77775 - 137757
4374                                                          ;TRY TO CAUSE AN ARITHMETIC OVERFLOW
4375  016220  102003                  BVC     1$                      ;ERROR IF V CLEAR
4376  016222  100002                  BPL     1$                      ;ERROR IF RESULT IS POSITIVE
4377  016224  001401                  BEQ     1$                      ;ERROR IF  Z SET
4378  016226  103401                  BCS     2$                      ;BRANCH IF GOOD
4379  016230  104001          1$:     ERROR   ·1              ;CPU ERROR
4380                                                          ;BAD SUBTRACT
4381  016232  012704  000005  2$:     MOV     #5,R4                   ;R4=5
4382  016236  000257                  CCC                             ;CC=0000
4383  016240  162704  000012          SUB     #12,R4                  ;5-12=-5 AND SETS CARRY
```

```
4384  016244  103003                    BCC     3$                      ;ERROR IF CARRY CLEAR
4385  016246  102402                    BVS     3$                      ;ERROR IF OVERFLOW
4386  016250  001401                    BEQ     3$                      ;ERROR IF ZERO
4387  016252  100401                    BMI     4$                      ;BRANCH IF GOOD
4388  016254  104001              3$:   ERROR   .1              ;CPU ERROR
4389                                                          ;SUBTRACT FAILED
4390  016256                        4$:
4391
4392
4393  016256                        MSBCCC:
4394
4395                                ;       TEST SBC CONDITION CODES
4396  016256  012704  100000               MOV     #100000,R4              ;R4=100000
4397  016262  000257                       CCC                             ;C=0000
4398  016264  005604                       SBC     R4                      ;TRY TO SET V
4399  016266  100006                       BPL     1$                      ;ERROR IF N CLEAR
4400  016270  102405                       BVS     1$                      ;ERROR IF V SET (HAVENT SET C YET)
4401  016272  000261                       SEC                             ;CC SHOULD = 1001
4402  016274  005604                       SBC     R4                      ;TRY AGAIN TO SET V
4403  016276  102002                       BVC     1$                      ;ERROR IF V CLEAR
4404  016300  103401                       BCS     1$                      ;ERROR IF C SET
4405  016302  100001                       BPL     2$                      ;BRANCH IF GOOD
4406  016304  104001              1$:   ERROR   .1              ;CPU ERROR
4407                                                          ;SBC FAILED
4408  016306  005004              2$:      CLR     R4                      ;R4=0
4409  016310  000277                       SCC
4410  016312  000241                       CLC                             ;CC=1110
4411  016314  005604                       SBC     R4                      ;TRY TO CAUSE C FLAG FAILURE
4412  016316  103410                       BCS     3$                      ;ERROR IF C SET
4413  016320  102407                       BVS     3$                      ;ERROR IF V SET
4414  016322  001006                       BNE     3$                      ;ERROR IF  NOT ZERO
4415  016324  000261                       SEC                             ;SET CARRY
4416  016326  005604                       SBC     R4                      ;NOW, 0 - CARRY = 177777
4417  016330  103003                       BCC     3$                      ;ERROR IF CARRY CLEAR
4418  016332  102402                       BVS     3$                      ;ERROR IF V SET
4419  016334  001401                       BEQ     3$                      ;ERROR IF ZERO
4420  016336  100401                       BMI     4$                      ;BRANCH IF GOOD
4421  016340  104001              3$:   ERROR   .1              ;CPU ERROR
4422                                                          ;SBC FAILED
4423  016342                        4$:
4424
4425
4426  016342                        MRLCC:
4427
4428                                ;       TEST ROL CONDITION CODES
4429  016342  012704  060000               MOV     #60000,R4               ;R4= 0110000000000000
4430  016346  000257                       CCC                             ;CC=0000
4431  016350  006104                       ROL     R4                      ;R4= 1100000000000000
4432  016352  103402                       BCS     1$                      ;ERROR IF CARRY
4433  016354  102001                       BVC     1$                      ;ERROR IF V CLEAR
4434  016356  100401                       BMI     2$                      ;BRANCH IF GOOD
4435  016360  104001              1$:   ERROR   .1              ;CPU ERROR
4436                                                          ;ROL  FAILED
4437  016362  006104              2$:      ROL     R4                      ;R4= 1000000000000000
4438  016364  103002                       BCC     3$                      ;ERROR IF CARRY CLEAR
4439  016366  102401                       BVS     3$                      ;ERROR IF V SET
```

```
4440  016370  100401                  BMI     4$                      ;BRANCH IF GOOD
4441  016372  104001          3$:     ERROR   ·1              ;CPU ERROR
4442                                                          ;BAU ROL
4443  016374  006104          4$:     ROL     R4                      ;R4 = 0000000000000001
4444  016376  102003                  BVC     5$                      ;ERROR IF V CLEAR
4445  016400  103002                  BCC     5$                      ;ERROR IF C CLEAR
4446  016402  100401                  BMI     5$                      ;ERROR IF MINUS
4447  016404  001001                  BNE     6$                      ;BRANCH IF GOOD
4448  016406  104001          5$:     ERROR   ·1              ;CPU ERROR
4449                                                          ;BAD ROL
4450  016410  006104          6$:     ROL     R4                      ;R4=000000000000011
4451  016412  102402                  BVS     7$                      ;ERROR IF V SET
4452  016414  103401                  BCS     7$                      ;ERROR IF C SET
4453  016416  100001                  BPL     8$                      ;BRANCH IF GOOD
4454  016420  104001          7$:     ERROR   ·1              ;CPU ERROR
4455                                                          ;BAD ROL
4456  016422                  8$:
4457
4458
4459  016422                  MRRCC:
4460
4461                          ;       TEST ROR CONDITION CODES
4462  016422  012704  000003                MOV     #3,R4                   ;R4= 0000000000000011
4463  016426  000257                        CCC                             ;CC= 0000
4464  016430  006004                        ROR     R4                      ;R4= 0000000000000001
4465  016432  103002                        BCC     1$                      ;ERROR IF NO CARRY
4466  016434  102001                        BVC     1$                      ;ERROR IF V CLEAR
4467  016436  100001                        BPL     2$                      ;BRANCH IF GOOD
4468  016440  104001          1$:     ERROR   ·1              ;CPU ERROR
4469                                                          ;ROR  FAILED
4470  016442  006004          2$:     ROR     R4                      ;R4= 1000000000000000
4471  016444  103002                  BCC     3$                      ;ERROR IF CARRY CLEAR
4472  016446  102401                  BVS     3$                      ;ERROR IF V SET
4473  016450  100401                  BMI     4$                      ;BRANCH IF GOOD
4474  016452  104001          3$:     ERROR   ·1              ;CPU ERROR
4475                                                          ;BAD ROR
4476  016454  006004          4$:     ROR     R4                      ;R4 = 1100000000000000
4477  016456  102002                  BVC     5$                      ;ERROR IF V
4478  016460  103401                  BCS     5$                      ;ERROR IF C SET
4479  016462  100401                  BMI     6$                      ;BRANCH IF GOOD
4480  016464  104001          5$:     ERROR   ·1              ;CPU ERROR
4481                                                          ;BAD ROR
4482  016466  006004          6$:     ROR     R4                      ;R4= 0110000000000000
4483  016470  102402                  BVS     7$                      ;ERROR IF V SET
4484  016472  103401                  BCS     7$                      ;ERROR IF C SET
4485  016474  100001                  BPL     8$                      ;BRANCH IF GOOD
4486  016476  104001          7$:     ERROR   ·1              ;CPU ERROR
4487                                                          ;BAD ROR
4488  016500                  8$:
4489
4490
4491  016500                  XCBIT:
4492
4493                          ;       TEST C BIT WITH ROR/ROL
4494                          ;;**************************************************************
4495                          ;THIS TEST IS TO CHECK FOR A SLOW C BIT PATH INTERNAL TO THE J11 DATA CHIP
```

```
                        ************ DOUBLE OPERAND TESTS ******************                    SEQ 0084

                        ;PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 AND 1593)
016500  012701  052525          MOV     #52525,R1               ;INIT R1 WITH DATA
016504  000241                  CLC                             ;CLEAR THE C BIT
016506  006001                  ROR     R1                      ;R1=025252, C BIT =1
016510  006001                  ROR     R1                      ;R1=112525, C BIT =0
016512  006001                  ROR     R1                      ;R1=045252, C BIT =1
016514  103401                  BCS     1$                      ;BRANCH IF CARRY BIT SET
016516  104001                  ERROR   +1              ;CPU ERROR
016520  022701  045252  1$:     CMP     #45252,R1               ;IS DATA IN R1 = TO EXPECTED DATA?
016524  001401                  BEQ     2$                      ;BRANCH IF YES
016526  104001                  ERROR   +1              ;CPU ERROR
016530  012701  125252  2$:     MOV     #125252,R1              ;SET UP R1
016534  000241                  CLC                             ;CLEAR THE CARRY BIT
016536  006101                  ROL     R1                      ;R1=052524, C BIT =1
016540  006101                  ROL     R1                      ;R1=125251, C BIT =0
016542  006101                  ROL     R1                      ;R1=052522, C BIT =1
016544  103401                  BCS     3$                      ;BRANCH IF CARRY SET
016546  104001                  ERROR   +1              ;CPU ERROR
016550  022701  052522  3$:     CMP     #052522, R1             ;IS DATA IN R1 = TO EXPECTED DATA?
016554  001401                  BEQ     4$
016556  104001                  ERROR   +1              ;CPU ERROR
016560                  4$:

016560                  MALCC:

                        ;       TEST ASL CONDITION CODES
016560  012704  060000          MOV     #60000,R4               ;R4= 0110000000000000
016564  000257                  CCC                             ;CC=0000
016566  006304                  ASL     R4                      ;C=0 R4= 1100000000000000
016570  103402                  BCS     1$                      ;ERROR IF CARRY
016572  102001                  BVC     1$                      ;ERROR IF V CLEAR
016574  100401                  BMI     2$                      ;BRANCH IF GOOD
016576  104001          1$:     ERROR   +1              ;CPU ERROR
                                                        ;ASL  FAILED
016600  006304          2$:     ASL     R4                      ;C=1 R4= 1000000000000000
016602  103002                  BCC     3$                      ;ERROR IF CARRY CLEAR
016604  102401                  BVS     3$                      ;ERROR IF V SET
016606  100401                  BMI     4$                      ;BRANCH IF GOOD
016610  104001          3$:     ERROR   +1              ;CPU ERROR
                                                        ;BAD ASL
016612  006304          4$:     ASL     R4                      ;C=1 R4= 0000000000000000
016614  102003                  BVC     5$                      ;ERROR IF V CLEAR
016616  103002                  BCC     5$                      ;ERROR IF C CLEAR
016620  100401                  BMI     5$                      ;ERROR IF MINUS
016622  001401                  BEQ     6$                      ;BRANCH IF GOOD
016624  104001          5$:     ERROR   +1              ;CPU ERROR
                                                        ;BAD ASL
016626  006304          6$:     ASL     R4                      ;C=0 R4= 0000000000000000
016630  102402                  BVS     7$                      ;ERROR IF V SET
016632  103401                  BCS     7$                      ;ERROR IF C SET
016634  100001                  BPL     8$                      ;BRANCH IF GOOD
016636  104001          7$:     ERROR   +1              ;CPU ERROR
                                                        ;BAD ASL
016640                  8$:

016640                  MARCC:
```

```
                                ;       TEST ASR CONDITION CODES
016640  012704  000341                  MOV     #341,R4                 ;R4= 0000000011100001
016644  000257                          CCC                             ;CC=0000
016646  006204                          ASR     R4                      ;R4= 0000000001110000
016650  103002                          BCC     1$                      ;ERROR IF NO CARRY
016652  102001                          BVC     1$                      ;ERROR IF V CLEAR
016654  100001                          BPL     2$                      ;BRANCH IF GOOD
016656  104001                  1$:     ERROR   ;1              ;CPU ERROR
                                                                ;ASR  FAILED
016660  052704  100001          2$:     BIS     #100001,R4              ;R4= 1000000001110001
016664  006204                          ASR     R4                      ;R4= 1100000000111000
016666  103002                          BCC     3$                      ;ERROR IF CARRY CLEAR
016670  102401                          BVS     3$                      ;ERROR IF V SET
016672  100401                          BMI     4$                      ;BRANCH IF GOOD
016674  104001                  3$:     ERROR   ;1              ;CPU ERROR
                                                                ;BAD ASR
016676  006204                  4$:     ASR     R4                      ;R4= 1110000000011100
016700  102002                          BVC     5$                      ;ERROR IF V
016702  103401                          BCS     5$                      ;ERROR IF C SET
016704  100401                          BMI     6$                      ;BRANCH IF GOOD
016706  104001                  5$:     ERROR   ;1              ;CPU ERROR
                                                                ;BAD ASR
016710  006204                  6$:     ASR     R4                      ;R4= 1111000000001110
016712  102005                          BVC     7$                      ;ERROR IF V CLEAR
016714  103404                          BCS     7$                      ;ERROR IF C SET
016716  100003                          BPL     7$                      ;ERROR IF PLUS
016720  022704  170016                  CMP     #170016,R4              ;SEE IF EXPECTED RESULT
016724  001401                          BEQ     8$                      ;BRANCH IF GOOD
016726  104001                  7$:     ERROR   ;1              ;CPU ERROR
                                                                ;BAD ASR
016730                          8$:

016730                          MSXTCC:

                                ;       TEST SXT CONDITION CODES / -*0-
016730  012704  123456                  MOV     #123456,R4                      ;R4=123456
016734  010401                          MOV     R4,R1                   ;SAVE  CONTENTS
016736  000257                          CCC                             ;CC=0000
016740  006704                          SXT     R4                      ;R4=0 CC=0100
016742  103403                          BCS     1$                      ;ERROR IF CARRY
016744  100402                          BMI     1$                      ;ERROR IF MINUS
016746  102401                          BVS     1$                      ;ERROR IF OVERFLOW
016750  001401                          BEQ     2$                      ;BRANCH IF GOOD
016752  104001                  1$:     ERROR   ;1              ;CPU ERROR
                                                                ;BAD SXT
016754  010104                  2$:     MOV     R1,R4                   ;RESTORE R4
016756  000277                          SCC                             ;CC=1111
016760  006704                          SXT     R4                      ;R4=-1 CC=1001
016762  001405                          BEQ     3$                      ;ERROR IF ZERO
016764  100004                          BPL     3$                      ;ERROR IF PLUS
016766  103003                          BCC     3$                      ;ERROR IF NO CARRY
016770  102402                          BVS     3$                      ;ERROR IF OVERFLOW
016772  005104                          COM     R4                      ;R4=0
016774  001401                          BEQ     4$                      ;BRANCH IF GOOD
```

```
4608  016776  104001                  3$:     ERROR   ^1              ;CPU ERROR
4609                                                                  ;BAD SXT
4610  017000                          4$:
4611
4612
4613  017000                          MXRCC:
4614
4615                                  ;       TEST XOR CONDITION CODES / **0-
4616  017000  012704  123456                  MOV     #123456,R4              ;R4=123456
4617  017004  012701  052525                  MOV     #52525,R1               ;R1=52525
4618  017010  000257                          CCC                             ;CC=0000
4619  017012  074104                          XOR     R1,R4                   ;*TI* R4=171173
4620  017014  102403                          BVS     1$                      ;;ERROR IF OVERFLOW
4621  017016  001402                          BEQ     1$                      ;ERROR IF ZERO
4622  017020  103401                          BCS     1$                      ;ERROR IF CARRY
4623  017022  100401                          BMI     2$                      ;BRANCH IF GOOD
4624  017024  104001                  1$:     ERROR   ^1              ;CPU ERROR
4625                                                                  ;BAD XOR
4626  017026  012701  125252          2$:     MOV     #125252,R1              ;R1=125252
4627  017032  000277                          SCC                             ;CC=1111
4628  017034  074104                          XOR     R1,R4                   ;R4=054321
4629  017036  100403                          BMI     3$                      ;ERROR IF MINUS
4630  017040  001402                          BEQ     3$                      ;ERROR IF ZERO
4631  017042  103001                          BCC     3$                      ;ERROR IF CARRY CLEAR
4632  017044  102001                          BVC     4$                      ;BRANCH IF GOOD
4633  017046  104001                  3$:     ERROR   ^1              ;CPU ERROR
4634                                                                  ;BAD XOR
4635  017050  074404                  4$:     XOR     R4,R4                   ;R4=0
4636  017052  102406                          BVS     5$                      ;ERROR IF OVREFLOW
4637  017054  100405                          BMI     5$                      ;ERROR IF MINUS
4638  017056  103004                          BCC     5$                      ;ERROR IF NO CARRY
4639  017060  001003                          BNE     5$                      ;ERROR IF NOT ZERO
4640  017062  022704  000000                  CMP     #0,R4                   ;SEE IF EXPECTED RESULT
4641  017066  001401                          BEQ     6$                      ;BRANCH IF GOOD
4642  017070  104001                  5$:     ERROR   ^1              ;CPU ERROR
4643                                                                  ;BAD XOR
4644  017072                          6$:
4645
4646
4647                                  ;
4648  017072                          MSXT:
4649
4650                                  ;       TEST SXT (SIGN EXTEND INSTRUCTION)
4651                                  ;;*****************************************************************
4652                                  ;AN ADDITIONAL TEST IS INCLUDED TO CHECK FOR A SLOW N BIT PATH
4653                                  ;ON A TRANSITION FROM ZERO TO ONE INTERNAL TO THE J11 DATA CHIP
4654                                  ;THE PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 OR 1593)
4655  017072  005004                          CLR     R4                      ;TRASH R4
4656  017074  000257                          CCC                             ;CC=0000
4657  017076  000271                          .WORD   271                     ;CC=1001
4658  017100  006704                          SXT     R4                      ;*TEST INSTRUCTION
4659  017102  102405                          BVS     1$                      ;BRANCH IF OVERFLOW IS NOT CLEARED
4660  017104  100004                          BPL     1$                      ;BRANCH IF N BIT EFFECTED
4661  017106  001403                          BEQ     1$                      ;BRANNCH IF Z BIT EFFECTED
4662  017110  103002                          BCC     1$                      ;BRANCH IF C BIT EFFECTED
4663  017112  005204                          INC     R4
```

```
                        ************ DOUBLE OPERAND TESTS ****************                SEQ 0087

 4664  017114  001401                        BEQ     2$              ;BRANCH IF R4 =0
 4665  017116  104001              1$:       ERROR   +1              ;CPU ERROR
 4666                                                  ;CC SHOULD HAVE = 1101
 4667  017120  000277              2$:       SCC
 4668  017122  000250                        CLN                     ;CC=0111
 4669  017124  005004                        CLR     R4
 4670  017126  012714  000055              MOV     #55,(R4)        ;TRASH R4
 4671  017132  006714                        SXT     (R4)            ;*TEST INSTRUCTION
 4672  017134  001005                        BNE     3$              ;BRANCH IF BIT EFFECTED
 4673  017136  102404                        BVS     3$              ;BRANCH IF OVERFLOW
 4674  017140  103403                        BCS     3$              ;
 4675  017142  100402                        BMI     3$              ;BRANCH IF N IS SET
 4676  017144  005714                        TST     (R4)            ;VERIFY INSTRUCTION WORKED
 4677  017146  001401                        BEQ     4$              ;BRANCH IF R4=0
 4678  017150  104001              3$:       ERROR   +1              ;CPU ERROR
 4679                                                  ;SXT FAILED
 4680
 4681                              ;         NOW TEST FOR SLOW N BIT IN J11 DATA CHIP
 4682
 4683  017152  012700  177777      4$:       MOV     #-1,R0          ;R0=177777, N BIT = 1
 4684  017156  005004                        CLR     R4              ;CLEAT THE N BIT
 4685  017160  006700                        SXT     R0              ;****TEST INSTRUCTION***
 4686                                                                ;TEST N BIT TRANSITION 1 TO 0
 4687  017162  005700                        TST     R0              ;R0 SHOULD = 0
 4688  017164  001401                        BEQ     5$              ;BRANCH IF OK
 4689  017166  104001                        ERROR   +1              ;CPU ERROR
 4690  017170  005000              5$:       CLR     R0              ;CLEAR R0, N BIT = 0
 4691  017172  012704  177777              MOV     #-1,R4          ;SET N BIT
 4692  017176  006700                        SXT     R0              ;***TEST INSTRUCTION***
 4693                                                                ;TEST N BIT TRANSITION 0 TO 1
 4694  017200  022700  177777              CMP     #-1,R0          ;R0 SHOULD = 177777
 4695  017204  001401                        BEQ     6$              ;BRANCH IF OK
 4696  017206  104001                        ERROR   +1              ;CPU ERROR
 4697  017210                      6$:
 4698                              ;
 4699  017210                      MXOR:
 4700
 4701                              ;         TEST XOR
 4702  017210  012701  007643              MOV     #7643,R1                ;SETUP DATA
 4703  017214  012704  133333              MOV     #133333,R4              ;SETUP DATA
 4704  017220  000277                        SCC
 4705  017222  074401                        XOR     R4,R1                   ;*TEST INSTRUCTION
 4706  017224  100006                        BPL     1$                      ;BRANCH IF PLUS TO ERROR
 4707  017226  001405                        BEQ     1$                      ;ERROR IF ZERO
 4708  017230  103004                        BCC     1$                      ;ERROR IF CARRY CLEAR
 4709  017232  102403                        BVS     1$                      ;ERROR IF V SET
 4710  017234  020127  134570              CMP     R1,#134570              ;VERIFY CORRECT RESULT
 4711  017240  001401                        BEQ     2$                      ;BRANCH IF GOOD
 4712  017242  104001              1$:       ERROR   +1              ;CPU ERROR
 4713                                                                ;BAD XOR
 4714  017244  010102              2$:       MOV     R1,R2
 4715  017246  000257                        CCC
 4716  017250  074402                        XOR     R4,R2                   ;*TEST INSTRUCTION
 4717  017252  100405                        BMI     3$                      ;ERROR IF MINUS
 4718  017254  102404                        BVS     3$                      ;ERROR IF OVERFLOW
 4719  017256  103403                        BCS     3$                      ;ERROR IF CARRY
```

```
 4720  017260  020227  007643                  CMP     R2,#7643
 4721  017264  001401                          BEQ     4$                      ;BRANCH IF GOOD
 4722  017266  104001                  3$:     ERROR   .1                      ;CPU ERROR
 4723                                                                          ;BAD XOR
 4724  017270                          4$:
 4725
 4726                                  ;
 4727  017270                          MSOB:
 4728
 4729                                  ;       TEST SOB
 4730  017270  012704  000555                  MOV     #555,R4                 ;SETUP TEST COUNTER
 4731  017274  000277                          SCC                             ;CC=17
 4732  017276  103015                  1$:     BCC     2$                      ;ERROR IF CARRY CLEAR
 4733  017300  102014                          BVC     2$                      ;ERROR IF NO OVERFLOW
 4734  017302  100013                          BPL     2$                      ;ERROR IF PLUS
 4735  017304  001012                          BNE     2$                      ;ERROR IF ZERO
 4736  017306  077405                          SOB     R4,1$                   ;*TEST INSTRUCTION
 4737  017310  103005                          BCC     3$                      ;ERROR IF CARRY CLEAR
 4738  017312  102004                          BVC     3$                      ;ERROR IF NO OVERFLOW
 4739  017314  100003                          BPL     3$                      ;ERROR IF PLUS
 4740  017316  001002                          BNE     3$                      ;ERROR IF ZERO
 4741  017320  000167  000010                  JMP     4$
 4742  017324  104001                  3$:     ERROR   .1                      ;CPU ERROR
 4743                                                                  ;CC EFFECTED DURING TEST
 4744  017326  000167  000002                  JMP     4$
 4745  017332  104001                  2$:     ERROR   .1                      ;CPU ERROR
 4746                                                                  ;CC EFFECTED AFTER TEST
 4747  017334  020427  000000          4$:     CMP     R4,#0                   ;IS R4 CORRECT
 4748  017340  001401                          BEQ     5$                      ;YES GO ON
 4749  017342  104001                          ERROR   .1                      ;CPU ERROR
 4750                                                                  ;NO GO TO ERROR
 4751  017344                          5$:
 4752
 4753
 4754  017344                          MMARK:
 4755
 4756                                  ;       MARK INSTRUCTION TEST
 4757  017344  012706  000700                  MOV     #STBOT-100,SP           ;SETUP TEST STACK = 700
 4758  017350  012737  125252  000776          MOV     #125252,@#STBOT-2       ;SET UP NEW R5 VALUE ON STACK
 4759  017356  012705  017374                  MOV     #1$,R5                  ;PUT NEW PC IN R5
 4760  017362  012746  006437                  MOV     #MARK+37,-(SP)          ;INSERT MARK 37 INSTRUCTION ONTO STACK
 4761  017366  000277                          SCC
 4762  017370  000116                          JMP     (SP)                    ;* TEST INSTRUCTION
 4763                                                                          ;MARK INSTRUCTION SHOULD HAVE GONE TO 1$
 4764  017372  104001                          ERROR   .1              ;CPU ERROR
 4765
 4766  017374  101002                  1$:     BHI     2$                      ;ERROR IF C OR Z BIT CLEAR
 4767  017376  100001                          BPL     2$                      ;ERROR IF N BIT CLEAR
 4768  017400  102401                          BVS     3$                      ;BRANCH IF V BIT SET
 4769                                                                          ;BAD CONDITION CODES ON MARK
 4770  017402  104001                  2$:     ERROR   .1              ;CPU ERROR
 4771  017404  022705  125252          3$:     CMP     #125252,R5              ;VERIFY R5
 4772  017410  001401                          BEQ     4$                      ;BRANCH IF GOOD
 4773  017412  104001                          ERROR   .1              ;CPU ERROR
 4774
 4775  017414  020627  001000          4$:     CMP     SP,#STBOT               ;VERIFY THAT STACK IS CORRECT
```

```
 4776  017420  001401                  BEQ     15$                     ;BRANCH IF OK
 4777                                                                  ;ERROR! STACK WAS NOT CORRECT AFTER MARK
 4778  017422  104001                  ERROR   .1              ;CPU ERROR
 4779
 4780  017424  012746  052525  15$:    MOV     #52525,-(SP)            ;SETUP EXPECTED R5
 4781  017430  012746  006400          MOV     #6400,-(SP)             ;MOVE MARK 0 INSTRUCTION ON STACK
 4782  017434  010605                  MOV     SP,R5                   ;R5=ADDRESS OF INSTRUCTION
 4783  017436  004767  000004          JSR     PC,5$                   ;LEAVE 6$ ON STACK
 4784  017442  000167  000006  6$:     JMP     16$                     ;MARK RETURNED CORRECTLY
 4785
 4786  017446  000257          5$:     CCC                             ;CLEAR THE CONDITION CODES
 4787  017450  000205                  RTS     R5                      ;RETURN TO MARK INSTRUCTION
 4788                                                                  ;NEXT INSTRUCTION ON STACK IS THE RETURN
 4789                                                                  ;FROM THE JSR
 4790                                                                  ;ERROR! BAD MARK SEQUENCE
 4791  017452  104001                  ERROR   .1              ;CPU ERROR
 4792
 4793  017454  101402          16$:    BLOS    7$                      ;IS C OR Z BIT SET?
 4794  017456  100401                  BMI     7$                      ;IS N BIT SET?
 4795  017460  102001                  BVC     8$                      ;IS V BIT SET?
 4796                                                                  ;ERROR! CONDITIONS CODES INCORRECT
 4797  017462  104001          7$:     ERROR   .1              ;CPU ERROR
 4798
 4799  017464  020627  001000  8$:     CMP     R6,#STBOT               ;IS THE STACK CORRECT?
 4800  017470  001401                  BEQ     9$                      ;BRANCH IF YES
 4801                                                                  ;ERROR! BAD STACK CLEANUP
 4802  017472  104001                  ERROR   .1              ;CPU ERROR
 4803  017474  022705  052525  9$:     CMP     #52525,R5               ;VERIFY THAT R5 WAS LOADED PROPERLY.
 4804  017500  001401                  BEQ     10$                     ;IF OK, GO TO NEXT TEST.
 4805                                                                  ;ERROR! R5 WAS NOT CORRECT AFTER MARK.
 4806  017502  104001                  ERROR   .1              ;CPU ERROR
 4807  017504                  10$:
 4808  017504                  XME100:
 4809
 4810                          ;       TEST CLEAR CONDITION CODES INSTRUCTION
 4811  017504  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
 4812  017512  000257                          CCC                             ; TEST INSTRUCTION
 4813  017514  022737  030000  177776          CMP     #30000,@#177776         ;DID IT CLEAR ALL CONDITION CODE BITS
 4814  017522  001401                          BEQ     1$                      ;YES GO ON
 4815  017524  104001                          ERROR   .1              ;CPU ERROR
 4816                                                                  ;NO GO TO ERROR
 4817  017526                  1$:
 4818                          ;
 4819  017526                  XME101:
 4820
 4821                          ;       TEST CLEAR C BIT INSTRUCTION
 4822  017526  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
 4823  017534  000241                          CLC                             ; TEST INSTRUCTION
 4824  017536  022737  030016  177776          CMP     #30016,@#177776         ;DID IT CLEAR CARRY BIT
 4825  017544  001401                          BEQ     1$                      ;YES GO ON
 4826                                                                          ;C BIT NOT CLEAR GO TO ERROR
 4827  017546  104001                          ERROR   .1              ;CPU ERROR
 4828  017550                  1$:
 4829                          ;
 4830  017550                  TE102:
 4831                          ;       TEST CLEAR N BIT INST (CLN)
```

```
4832  017550  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4833  017556  000250                          CLN                             ; TEST INSTRUCTION
4834  017560  022737  030007  177776          CMP     #30007,@#177776         ;DID IT CLEAR NEGATIVE BIT
4835  017566  001401                          BEQ     1$                      ;YES GO ON
4836  017570  104001                          ERROR   .+1             ;CPU ERROR
4837                                                                  ;NO GO TO ERROR
4838  017572                          1$:
4839                                  ;
4840  017572                          TE103:
4841                                  ;       TEST CLEAR V BIT INST (CLV)
4842  017572  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4843  017600  000242                          CLV                             ; TEST INSTRUCTION
4844  017602  022737  030015  177776          CMP     #30015,@#177776         ;DID IT CLEAR OVERFLOW BIT
4845  017610  001401                          BEQ     1$                      ;YES GO ON
4846  017612  104001                          ERROR   .+1             ;CPU ERROR
4847                                                                  ;NO GO TO ERROR
4848  017614                          1$:
4849                                  ;
4850  017614                          TE104:
4851                                  ;       TEST CLEAR Z BIT INST (CLZ)
4852  017614  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4853  017622  000244                          CLZ                             ; TEST INSTRUCTION
4854  017624  022737  030013  177776          CMP     #30013,@#177776         ;DID IT CLEAR ZERO BIT
4855  017632  001401                          BEQ     1$                      ;YES GO ON
4856  017634  104001                          ERROR   .+1             ;CPU ERROR
4857                                                                  ;NO GO TO ERROR
4858  017636                          1$:
4859                                  ;
4860  017636                          TE105:
4861                                  ;       TEST SET CONDITION CODES INST (SCC)
4862  017636  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
4863  017644  000277                          SCC                             ; TEST INSTRUCTION
4864  017646  022737  030017  177776          CMP     #30017,@#177776         ;DID IT SET ALL CONDITION CODE BITS
4865  017654  001401                          BEQ     1$                      ;YES GO ON
4866  017656  104001                          ERROR   .+1             ;CPU ERROR
4867                                                                  ;NO GO TO ERROR
4868  017660                          1$:
4869                                  ;
4870  017660                          TE106:
4871                                  ;       TEST SET C BIT INST (SEC)
4872  017660  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
4873  017666  000261                          SEC                             ; TEST INSTRUCTION
4874  017670  022737  030001  177776          CMP     #30001,@#177776         ;DID IT SET THE CARRY BIT
4875  017676  001401                          BEQ     1$                      ;YES GO ON
4876  017700  104001                          ERROR   .+1             ;CPU ERROR
4877                                                                  ;NO GO TO ERROR
4878  017702                          1$:
4879                                  ;
4880  017702                          TE107:
4881                                  ;       TEST SET N BIT INST (SEN)
4882  017702  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
4883  017710  000270                          SEN                             ; TEST INSTRUCTION
4884  017712  022737  030010  177776          CMP     #30010,@#177776         ;DID IT SET THE NEGATIVE BIT
4885  017720  001401                          BEQ     1$                      ;YES GO ON
4886  017722  104001                          ERROR   .+1             ;CPU ERROR
4887                                                                  ;NO GO TO ERROR
```

```
 4888  017724                          1$:
 4889                                  ;
 4890  017724                          TE110:
 4891                                  ;       TEST SET V BIT INST (SEV)
 4892  017724  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 4893  017732  000262                          SEV                             ; TEST INSTRUCTION
 4894  017734  022737  030002  177776          CMP     #30002,@#177776         ;DID IT SET THE OVERFLOW BIT
 4895  017742  001401                          BEQ     1$                      ;YES GO ON
 4896  017744  104001                          ERROR   +1              ;CPU ERROR
 4897                                                                  ;NO GO TO ERROR
 4898  017746                          1$:
 4899                                  ;
 4900  017746                          TE111:
 4901                                  ;       TEST SET Z BIT INST (SEZ)
 4902  017746  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 4903  017754  000264                          SEZ                             ; TEST INSTRUCTION
 4904  017756  022737  030004  177776          CMP     #30004,@#177776         ;DID IT SET THE ZERO BIT
 4905  017764  001401                          BEQ     1$                      ;YES GO ON
 4906  017766  104001                          ERROR   +1              ;CPU ERROR
 4907                                                                  ;NO GO TO ERROR
 4908  017770                          1$:
 4909                                  ;
 4910  017770                          TE112:
 4911                                  ;       TEST MULTIPLE CLEARS OF CC BITS
 4912  017770  012737  030000  177776          MOV     #30000,@#177776         ;INIT PSW
 4913  017776  000277                          SCC                             ;SETUP PSW
 4914  020000  000243                          .WORD   243                     ; TEST CLC CLV
 4915  020002  022737  030014  177776          CMP     #30014,@#177776         ;PSW CORRECT?
 4916  020010  001401                          BEQ     1$                      ;YES GO ON
 4917  020012  104001                          ERROR   +1              ;CPU ERROR
 4918                                                                  ;NO GO TO ERROR
 4919  020014  000277                  1$:     SCC                             ;SETUP PSW
 4920  020016  000245                          .WORD   245                     ; TEST CLC CLZ
 4921  020020  022737  030012  177776          CMP     #30012,@#177776         ;PSW CORRECT?
 4922  020026  001401                          BEQ     2$                      ;YES GO ON
 4923  020030  104001                          ERROR   +1              ;CPU ERROR
 4924                                                                  ;NO GO TO ERROR
 4925  020032  000277                  2$:     SCC                             ;SETUP PSW
 4926  020034  000246                          .WORD   246                     ; TEST CLV CLZ
 4927  020036  022737  030011  177776          CMP     #30011,@#177776         ;PSW CORRECT?
 4928  020044  001401                          BEQ     3$                      ;YES GO ON
 4929  020046  104001                          ERROR   +1              ;CPU ERROR
 4930                                                                  ;NO GO TO ERROR
 4931  020050  000277                  3$:     SCC                             ;SETUP PSW
 4932  020052  000247                          .WORD   247                     ; TEST CLC CLV CLZ
 4933  020054  022737  030010  177776          CMP     #30010,@#177776         ;PSW CORRECT?
 4934  020062  001401                          BEQ     4$                      ;YES GO ON
 4935  020064  104001                          ERROR   +1              ;CPU ERROR
 4936                                                                  ;NO GO TO ERROR
 4937  020066  000277                  4$:     SCC                             ;SETUP PSW
 4938  020070  000251                          .WORD   251                     ; TEST CLN CLC
 4939  020072  022737  030006  177776          CMP     #30006,@#177776         ;PSW CORRECT?
 4940  020100  001401                          BEQ     5$                      ;YES GO ON
 4941  020102  104001                          ERROR   +1              ;CPU ERROR
 4942                                                                  ;NO GO TO ERROR
 4943  020104  000277                  5$:     SCC                             ;SETUP PSW
```

```
                        ;********** DOUBLE OPERAND TESTS ****************                SEQ 0092

4944  020106  000252                .WORD   252                     ; TEST CLN CLV
4945  020110  022737  030005  177776        CMP     #30005,@#177776         ;PSW CORRECT?
4946  020116  001401                BEQ     6$                      ;YES GO ON
4947  020120  104001                ERROR   .1              ;CPU ERROR
4948                                                      ;NO GO TO ERROR
4949  020122  000277          6$:   SCC                             ;SETUP PSW
4950  020124  000253                .WORD   253                     ; TEST CLN CLC CLV
4951  020126  022737  030004  177776        CMP     #30004,@#177776         ;PSW CORRECT?
4952  020134  001401                BEQ     7$                      ;YES GO ON
4953  020136  104001                ERROR   .1              ;CPU ERROR
4954                                                      ;NO GO TO ERROR
4955  020140  000277          7$:   SCC                             ;SETUP PSW
4956  020142  000254                .WORD   254                     ; TEST CLN CLZ
4957  020144  022737  030003  177776        CMP     #30003,@#177776         ;PSW CORRECT?
4958  020152  001401                BEQ     8$                      ;YES GO ON
4959  020154  104001                ERROR   .1              ;CPU ERROR
4960                                                      ;NO GO TO ERROR
4961  020156  000277          8$:   SCC                             ;SETUP PSW
4962  020160  000255                .WORD   255                     ; TEST CLN CLC CLZ
4963  020162  022737  030002  177776        CMP     #30002,@#177776         ;PSW CORRECT?
4964  020170  001401                BEQ     9$                      ;YES GO ON
4965  020172  104001                ERROR   .1              ;CPU ERROR
4966                                                      ;NO GO TO ERROR
4967  020174  000277          9$:   SCC                             ;SETUP PSW
4968  020176  000256                .WORD   256                     ; TEST CLN CLV CLZ
4969  020200  022737  030001  177776        CMP     #30001,@#177776         ;SETUP PSW
4970  020206  001401                BEQ     10$                     ;YES GO ON
4971  020210  104001                ERROR   .1              ;CPU ERROR
4972                                                      ;NO GO TO ERROR
4973  020212                    10$:
4974                              ;
4975  020212                    TE113:
4976                              ;       TEST MULTIPLE SETS OF CC BITS
4977  020212  012737  030000  177776        MOV     #30000,@#177776         ;INIT PSW
4978  020220  000263                .WORD   263                     ; TEST SEC SEV
4979  020222  022737  030003  177776        CMP     #30003,@#177776         ;PSW CORRECT?
4980  020230  001401                BEQ     1$                      ;YES GO ON
4981  020232  104001                ERROR   .1              ;CPU ERROR
4982                                                      ;NO GO TO ERROR
4983  020234  000257          1$:   CCC                             ;SETUP PSW
4984  020236  000265                .WORD   265                     ; TEST SEC SEZ
4985  020240  022737  030005  177776        CMP     #30005,@#177776         ;PSW CORRECT?
4986  020246  001401                BEQ     2$                      ;YES GO ON
4987  020250  104001                ERROR   .1              ;CPU ERROR
4988                                                      ;NO GO TO ERROR
4989  020252  000257          2$:   CCC                             ;SETUP PSW
4990  020254  000266                .WORD   266                     ; TEST SEV SEZ
4991  020256  022737  030006  177776        CMP     #30006,@#177776         ;PSW CORRECT?
4992  020264  001401                BEQ     3$                      ;YES GO ON
4993  020266  104001                ERROR   .1              ;CPU ERROR
4994                                                      ;NO GO TO ERROR
4995  020270  000257          3$:   CCC                             ;SETUP PSW
4996  020272  000267                .WORD   267                     ; TEST SEC SEV SEZ
4997  020274  022737  030007  177776        CMP     #30007,@#177776         ;PSW CORRECT?
4998  020302  001401                BEQ     4$                      ;YES GO ON
4999  020304  104001                ERROR   .1              ;CPU ERROR
```

```
5000
5001  020306  000257                  4$:     CCC                             ;NO GO TO ERROR
5002  020310  000271                          .WORD   271                             ;SETUP PSW
5003  020312  022737  030011  177776          CMP     #30011,@#177776                 ; TEST SEN SEC
5004  020320  001401                          BEQ     5$                              ;PSW CORRECT?
5005  020322  104001                          ERROR   .1                              ;YES GO ON
5006                                                                          ;CPU ERROR
5007  020324  000257                  5$:     CCC                             ;NO GO TO ERROR
5008  020326  000272                          .WORD   272                             ;SETUP PSW
5009  020330  022737  030012  177776          CMP     #30012,@#177776                 ; TEST SEN SEV
5010  020336  001401                          BEQ     6$                              ;PSW CORRECT?
5011  020340  104001                          ERROR   .1                              ;YES GO ON
5012                                                                          ;CPU ERROR
5013  020342  000257                  6$:     CCC                             ;NO GO TO ERROR
5014  020344  000273                          .WORD   273                             ;SETUP PSW
5015  020346  022737  030013  177776          CMP     #30013,@#177776                 ; TEST SEN SEC SEV
5016  020354  001401                          BEQ     7$                              ;PSW CORRECT?
5017  020356  104001                          ERROR   .1                              ;YES GO ON
5018                                                                          ;CPU ERROR
5019  020360  000257                  7$:     CCC                             ;NO GO TO ERROR
5020  020362  000274                          .WORD   274                             ;SETUP PSW
5021  020364  022737  030014  177776          CMP     #30014,@#177776                 ; TEST SEN SEZ
5022  020372  001401                          BEQ     8$                              ;PSW CORRECT?
5023  020374  104001                          ERROR   .1                              ;YES GO ON
5024                                                                          ;CPU ERROR
5025  020376  000257                  8$:     CCC                             ;NO GO TO ERROR
5026  020400  000275                          .WORD   275                             ;SETUP PSW
5027  020402  022737  030015  177776          CMP     #30015,@#177776                 ; TEST SEN SEC SEZ
5028  020410  001401                          BEQ     9$                              ;PSW CORRECT?
5029  020412  104001                          ERROR   .1                              ;YES GO ON
5030                                                                          ;CPU ERROR
5031  020414  000257                  9$:     CCC                             ;NO GO TO ERROR
5032  020416  000276                          .WORD   276                             ;SETUP PSW
5033  020420  022737  030016  177776          CMP     #30016,@#177776                 ; TEST SEN SEV SEZ
5034  020426  001401                          BEQ     10$                             ;PSW CORRECT?
5035  020430  104001                          ERROR   .1                              ;YES GO ON
5036                                                                          ;CPU ERROR
5037  020432                          10$:                                    ;NO GO TO ERROR
5038                                  ;
5039  020432                          TE113A:
5040                                  ;       TEST SIGNED AND CONDITIONAL BRANCHES
5041  020432  000257                          CCC                                     ;CLEAR ALL CC BITS IN PSW
5042  020434  002001                          BGE     1$                              ;BGE SHOULD BRANCH
5043  020436  104001                          ERROR   .1                              ;CPU ERROR
5044                                                                          ;ERROR; DIDN'T BRANCH
5045  020440  003001                  1$:     BGT     2$                              ;BGT SHOULD BRANCH
5046  020442  104001                          ERROR   .1                              ;CPU ERROR
5047                                                                          ;ERROR;  DIDN'T BRANCH
5048  020444  003401                  2$:     BLE     3$                              ;BLE SHOULDN'T BRANCH
5049  020446  000401                          BR      4$                              ;BRANCH TO NEXT TEST
5050  020450  104001                  3$:     ERROR   .1                              ;CPU ERROR
5051                                                                          ;ERROR; BLE SHOULD NOT HAVE BRANCHED
5052  020452  002401                  4$:     BLT     5$                              ;BLT SHOULD NOT BRANCH
5053  020454  000401                          BR      6$                              ;BRANCH TO NEXT TEST
5054  020456  104001                  5$:     ERROR   .1                              ;CPU ERROR
5055                                                                          ;ERROR; BLT SHOULD NOT HAVE BRANCHED
```

```
5056  020460  000264                  6$:     SEZ                             ;SET THE Z BIT IN PSW
5057  020462  003401                          BLE     7$                      ;BLE SHOULD BRANCH
5058  020464  104001                          ERROR   .1              ;CPU ERROR
5059                                                                  ;ERROR; BLE DIDN'T BRANCH
5060  020466  003001                  7$:     BGT     8$                      ;BGT SHOULD NOT BRANCH
5061  020470  000401                          BR      9$                      ;BRANCH TO NEXT TEST
5062  020472  104001                  8$:     ERROR   .1              ;CPU ERROR
5063                                                                  ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5064  020474  000257                  9$:     CCC                             ;CLEAR ALL CC BITS IN PSW
5065  020476  000270                          SEN                             ;SET N BIT IN PSW
5066  020500  002401                          BLT     10$                     ;SHOULD BRANCH TO NEXT TEST
5067  020502  104001                          ERROR   .1              ;CPU ERROR
5068                                                                  ;ERROR; BLT SHOULD HAVE BRANCHED
5069  020504  003401                  10$:    BLE     11$                     ;SHOULD BRANCH TO NEXT TEST
5070  020506  104001                          ERROR   .1              ;CPU ERROR
5071                                                                  ;ERROR; BLE SHOULD HAVE BRANCHED
5072  020510  002001                  11$:    BGE     12$                     ;BGE SHOULD NOT BRANCH
5073  020512  000401                          BR      13$                     ;BRANCH TO NEXT TEST
5074  020514  104001                  12$:    ERROR   .1              ;CPU ERROR
5075                                                                  ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5076  020516  003001                  13$:    BGT     14$                     ;BGT SHOULD NOT BRANCH
5077  020520  000401                          BR      15$                     ;BRANCH TO NEXT TEST
5078  020522  104001                  14$:    ERROR   .1              ;CPU ERROR
5079                                                                  ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5080  020524  000257                  15$:    CCC                             ;CLEAR ALL CC BITS
5081  020526  000262                          SEV                             ;SET V BIT IN PSW
5082  020530  003401                          BLE     16$                     ;BLE SHOULD BRANCH
5083  020532  104001                          ERROR   1               ;CPU ERROR
5084                                                                  ;ERROR; BLE DIDN'T BRANCH
5085  020534  002401                  16$:    BLT     17$                     ;BLT SHOULD BRANCH
5086  020536  104001                          ERROR   .1              ;CPU ERROR
5087                                                                  ;ERROR; BLT DIDN'T BRANCH
5088  020540  002001                  17$:    BGE     19$                     ;BGE SHOULDN'T BRANCH
5089  020542  000401                          BR      19$                     ;BRANCH TO NEXT TEST
5090  020544  104001                  18$:    ERROR   .1              ;CPU ERROR
5091                                                                  ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5092  020546  003001                  19$:    BGT     20$                     ;BGT SHOULD NOT BRANCH
5093  020550  000401                          BR      21$                     ;BRANCH TO NEXT TEST
5094  020552  104001                  20$:    ERROR   .1              ;CPU ERROR
5095                                                                  ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5096  020554  000257                  21$:    CCC                             ;CLEAR ALL CC BITS
5097  020556  000272                          .WORD   272                     ;SET N AND V BITS IN PSW
5098  020560  002001                          BGE     22$                     ;BGE SHOULD BRANCH
5099  020562  104001                          ERROR   .1              ;CPU ERROR
5100                                                                  ;ERROR; BGE DIDN'T BRANCH
5101  020564  003001                  22$:    BGT     23$                     ;BGT SHOULD BRANCH
5102  020566  104001                          ERROR   .1              ;CPU ERROR
5103                                                                  ;ERROR; BGT DIDN'T BRANCH
5104  020570  003401                  23$:    BLE     24$                     ;BLE SHOULDN'T BRANCH
5105  020572  000401                          BR      25$                     ;BRANCH TO NEXT TEST
5106  020574  104001                  24$:    ERROR   .1              ;CPU ERROR
5107                                                                  ;ERROR; BLE SHOULD NOT HAVE BRANCHED
5108  020576  002401                  25$:    BLT     26$                     ;BLT SHOULD NOT BRANCH
5109  020600  000401                          BR      27$                     ;BRANCH TO NEXT TEST
5110  020602  104001                  26$:    ERROR   .1              ;CPU ERROR
5111                                                                  ;ERROR; BLT SHOULD NOT HAVE BRANCHED
```

```
5112  020604                          27$:
5113  020604                          TE114:
5114                                  ;       TEST NOP INST
5115  020604  012737  030000  177776          MOV     #30000,@#177776         ;INIT PSW
5116  020612  012700  000001                  MOV     #1,R0                   ;INIT R0
5117  020616  012701  000002                  MOV     #2,R1                   ;INIT R1
5118  020622  012702  000003                  MOV     #3,R2                   ;INIT R2
5119  020626  012703  000004                  MOV     #4,R3                   ;INIT R3
5120  020632  012704  000005                  MOV     #5,R4                   ;INIT R4
5121  020636  012705  000006                  MOV     #6,R5                   ;INIT R5
5122  020642  010637  002762                  MOV     R6,@#SLOC00             ;SAVE SP
5123  020646  012737  030017  110150          MOV     #30017,@#EXPDAT         ;SETUP PSW
5124  020654  000277                          SCC                             ;SET ALL CONDITION CODE BITS
5125  020656  000240                          NOP                             ; TEST INSTRUCTION
5126  020660  000240                          NOP                             ; TEST INSTRUCTION
5127  020662  000240                          NOP                             ; TEST INSTRUCTION
5128  020664  004767  000046                  JSR     PC,T114                 ;CHECK PSW, AND GPR'S
5129  020670  020637  002762                  CMP     R6,@#SLOC00             ;CHECK SP
5130  020674  001401                          BEQ     1$                      ;OK GO ON
5131  020676  104001                          ERROR   .1              ;CPU ERROR
5132                                                                  ;NO GO TO ERROR
5133  020700  012737  030000  110150  1$:     MOV     #30000,@#EXPDAT         ;SETUP PSW
5134  020706  000257                          CCC                             ;CLEAR ALL CONDITION CODE BITS
5135  020710  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
5136  020712  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
5137  020714  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
5138  020716  004767  000014                  JSR     PC,T114                 ;CHECK PSW, AND GPR'S
5139  020722  020637  002762                  CMP     R6,@#SLOC00             ;CHECK SP
5140  020726  001401                          BEQ     2$                      ;OK GO ON
5141  020730  104001                          ERROR   .1              ;CPU ERROR
5142                                                                  ;NO GO TO ERROR
5143  020732                          2$:
5144  020732  000167  000104                  JMP     FINNOP
5145                                  ;
5146  020736  023737  110150  177776  T114:   CMP     @#EXPDAT,@#177776       ;CHECK PSW
5147  020744  001405                          BEQ     TA114                   ;OK GO ON
5148  020746  010067  157426                  MOV     R0,400                  ;SAVE R0
5149  020752  104001                          ERROR   .1              ;CPU ERROR
5150                                                                  ;NO GO TO ERROR
5151  020754  016700  157420                  MOV     400,R0                  ;RESTORE R0
5152  020760  022700  000001          TA114:  CMP     #1,R0                   ;CHECK R0
5153  020764  001401                          BEQ     TB114                   ;OK GO ON
5154  020766  104001                          ERROR   .1              ;CPU ERROR
5155                                                                  ;NO GO TO ERROR
5156  020770  022701  000002          TB114:  CMP     #2,R1                   ;CHECK R1
5157  020774  001401                          BEQ     TC114                   ;OK GO ON
5158  020776  104001                          ERROR   .1              ;CPU ERROR
5159                                                                  ;NO GO TO ERROR
5160  021000  022702  000003          TC114:  CMP     #3,R2                   ;CHECK R2
5161  021004  001401                          BEQ     TD114                   ;OK GO ON
5162  021006  104001                          ERROR   .1              ;CPU ERROR
5163                                                                  ;NO GO TO ERROR
5164  021010  022703  000004          TD114:  CMP     #4,R3                   ;CHECK R3
5165  021014  001401                          BEQ     TF114                   ;OK GO ON
5166  021016  104001                          ERROR   .1              ;CPU ERROR
5167                                                                  ;NO GO TO ERROR
```

```
5168  021020  022704  000005          TF114:  CMP     #5,R4                   ;CHECK R4
5169  021024  001401                          BEQ     TG114                   ;OK GO ON
5170  021026  104001                          ERROR   .+1             ;CPU ERROR
5171                                                                  ;NO GO TO ERROR
5172  021030  022705  000006          TG114:  CMP     #6,R5                   ;CHECK R5
5173  021034  001401                          BEQ     TH114                   ;OK GO ON
5174  021036  104001                          ERROR   .+1             ;CPU ERROR
5175                                                                  ;NO GO TO ERROR
5176  021040  000207                  TH114:  RTS     PC                      ;RETURN
5177                                  ;
5178  021042  000240                  FINNOP: NOP
5179                                  ;
5180                                  ;
5181                                  ;---------------------------------
5182                                  ;TEST ATERNATE REGISTER SET
5183                                  ;
5184  021044  005000                          CLR     R0                      ;------CLEAR------
5185  021046  005001                          CLR     R1                      ;-----PRIMARY----
5186  021050  005002                          CLR     R2                      ;-----GENERAL-----
5187  021052  005003                          CLR     R3                      ;-----PURPOSE-----
5188  021054  005004                          CLR     R4                      ;-----REGISTER----
5189  021056  005005                          CLR     R5                      ;-------SET-------
5190  021060  012767  004000  156710        MOV     #4000,PS                ;SELECT ALTERNATE REGISTER SET
5191  021066  032767  004000  156702        BIT     #BIT11,PS               ;TEST TO SEE THAT BIT IS SET IN PS
5192  021074  001001                          BNE     1$                      ;IF IT'S SET GO TESTS REGISTERS
5193  021076  104001                          ERROR   .+1             ;CPU ERROR
5194                                                                  ;ERROR! ALTERNATE REGISTER SELECT
5195                                                                  ;BIT IN PS IS NOT SET
5196  021100  012700  177777          1$:     MOV     #177777,R0              ;WRITE ALL ONES TO ALTERNATE REG SET
5197  021104  010001                          MOV     R0,R1                   ;
5198  021106  010102                          MOV     R1,R2                   ;
5199  021110  010203                          MOV     R2,R3                   ;
5200  021112  010304                          MOV     R3,R4                   ;
5201  021114  010405                          MOV     R4,R5                   ;
5202  021116  042767  004000  156652        BIC     #BIT11,PS               ;CLEAR BIT 11 IN PS
5203  021124  032767  004000  156644        BIT     #BIT11,PS               ;IS BIT 11 = 0?
5204  021132  001401                          BEQ     2$                      ;IF IT'S NOT ZERO
5205  021134  104001                          ERROR   .+1             ;CPU ERROR
5206                                                                  ;ERROR! BIT 11 DID NOT CLEAR
5207  021136  005700                  2$:     TST     R0                      ;MAKE SURE THAT WE REALLY
5208  021140  001401                          BEQ     3$                      ;WERE TALKING TO ALTERNATE REGISTERS.
5209  021142  104001                          ERROR   .+1             ;CPU ERROR
5210                                                                  ;ERROR!
5211  021144  005701                  3$:     TST     R1                      ;
5212  021146  001401                          BEQ     4$
5213  021150  104001                          ERROR   .+1             ;CPU ERROR
5214                                                                  ;ERROR!
5215  021152  005702                  4$:     TST     R2                      ;
5216  021154  001401                          BEQ     5$                      ;
5217  021156  104001                          ERROR   .+1             ;CPU ERROR
5218                                                                  ;ERROR!
5219  021160  005703                  5$:     TST     R3                      ;
5220  021162  001401                          BEQ     6$                      ;
5221  021164  104001                          ERROR   .+1             ;CPU ERROR
5222                                                                  ;
5223  021166  005704                  6$:     TST     R4                      ;
```

```
5224  021170  001401                        BEQ     7$                      ;
5225  021172  104001                        ERROR   .1              ;CPU ERROR
5226                                        ;
5227  021174  005705              7$:       TST     R5                      ;
5228  021176  001401                        BEQ     8$                      ;
5229  021200  104001                        ERROR   .1              ;CPU ERROR
5230                                        ;ERROR!
5231  021202  012767  004000  156566  8$:   MOV     #BIT11,PS               ;ENABLE ALTERNATE REGISTER SET
5232  021210  005000                        CLR     R0                      ;-----CLEAR-----
5233  021212  005001                        CLR     R1                      ;---ALTERNATE---
5234  021214  005002                        CLR     R2                      ;----REGISTER---
5235  021216  005003                        CLR     R3                      ;------SET-----
5236  021220  005004                        CLR     R4                      ;
5237  021222  005005                        CLR     R5                      ;
5238  021224  042767  004000  156544        BIC     #BIT11,PS               ;BACK TO PRIMARY GPRS
5239  021232  012700  177777                MOV     #177777,R0              ;ENSURE THAT WRITES TO PRIMARY GPRS
5240  021236  010001                        MOV     R0,R1                   ;DO NOT EFFECT ALTERNATE GPRS
5241  021240  010102                        MOV     R1,R2                   ;
5242  021242  010203                        MOV     R2,R3                   ;
5243  021244  010304                        MOV     R3,R4                   ;
5244  021246  010405                        MOV     R4,R5                   ;
5245  021250  052767  004000  156520        BIS     #BIT11,PS               ;RETURN TO ALTERNATE REGISTERS
5246  021256  005700                        TST     R0                      ;SHOULD BE ALL 0'S
5247  021260  001401                        BEQ     9$                      ;
5248  021262  104001                        ERROR   .1              ;CPU ERROR
5249                                        ;ERROR!
5250  021264  005701              9$:       TST     R1                      ;
5251  021266  001401                        BEQ     10$                     ;
5252  021270  104001                        ERROR   .1              ;CPU ERROR
5253                                        ;
5254  021272  005702             10$:       TST     R2                      ;
5255  021274  001401                        BEQ     11$                     ;
5256  021276  104001                        ERROR   .1              ;CPU ERROR
5257                                        ;
5258  021300  005703             11$:       TST     R3                      ;
5259  021302  001401                        BEQ     12$                     ;
5260  021304  104001                        ERROR   .1              ;CPU ERROR
5261                                        ;
5262  021306  005704             12$:       TST     R4                      ;
5263  021310  001401                        BEQ     13$                     ;
5264  021312  104001                        ERROR   .1              ;CPU ERROR
5265                                        ;
5266  021314  005705             13$:       TST     R5                      ;
5267  021316  001401                        BEQ     14$                     ;
5268  021320  104001                        ERROR   .1              ;CPU ERROR
5269  021322                    14$:
5270
5271  021322                    ALROTS:
5272                            ;           ALTERNATE REGISTER SET R0 BIT TESTS
5273  021322  012700  177777                MOV     #177777,R0              ;R0=177777
5274  021326  020027  177777                CMP     R0,#177777              ;DOES R0=177777
5275  021332  001401                        BEQ     1$                      ;YES GO ON
5276  021334  104001                        ERROR   .1              ;CPU ERROR
5277                                                                        ;NO GO TO ERROR
5278  021336  005000             1$:        CLR     R0                      ;R0=0
5279  021340  020027  000000                CMP     R0,#0                   ;DOES R0=0
```

```
5280  021344  001401                  BEQ     2$                      ;YES GO ON
5281  021346  104001                  ERROR   +1              ;CPU ERROR
5282                                                          ;NO GO TO ERROR
5283  021350  012700  125252  2$:     MOV     #125252,R0              ;R0=125252
5284  021354  020027  125252          CMP     R0,#125252              ;DOES R0=125252
5285  021360  001401                  BEQ     3$                      ;YES GO ON
5286  021362  104001                  ERROR   +1              ;CPU ERROR
5287                                                          ;NO GO TO ERROR
5288  021364  012700  052525  3$:     MOV     #52525,R0               ;R0=52525
5289  021370  020027  052525          CMP     R0,#52525               ;DOES R0=52525
5290  021374  001401                  BEQ     4$                      ;YES GO ON
5291  021376  104001                  ERROR   +1              ;CPU ERROR
5292                                                          ;NO GO TO ERROR
5293  021400                  4$:
5294                                  ;
5295  021400                  ALR1TS:
5296                          ;       ALTERNATE REGISTER SET R1 BIT TESTS
5297  021400  012701  177777          MOV     #177777,R1              ;R1=177777
5298  021404  020127  177777          CMP     R1,#177777              ;DOES R1=177777
5299  021410  001401                  BEQ     1$                      ;YES GO ON
5300  021412  104001                  ERROR   +1              ;CPU ERROR
5301                                                          ;NO GO TO ERROR
5302  021414  005001  1$:     CLR     R1                      ;R1=0
5303  021416  020127  000000          CMP     R1,#0                   ;DOES R1=0
5304  021422  001401                  BEQ     2$                      ;YES GO ON
5305  021424  104001                  ERROR   +1              ;CPU ERROR
5306                                                          ;NO GO TO ERROR
5307  021426  012701  125252  2$:     MOV     #125252,R1              ;R1=125252
5308  021432  020127  125252          CMP     R1,#125252              ;DOES R1=125252
5309  021436  001401                  BEQ     3$                      ;YES GO ON
5310  021440  104001                  ERROR   +1              ;CPU ERROR
5311                                                          ;NO GO TO ERROR
5312  021442  012701  052525  3$:     MOV     #52525,R1               ;R1=52525
5313  021446  020127  052525          CMP     R1,#52525               ;DOES R1=52525
5314  021452  001401                  BEQ     4$                      ;YES GO ON
5315  021454  104001                  ERROR   +1              ;CPU ERROR
5316                                                          ;NO GO TO ERROR
5317  021456                  4$:
5318                                  ;
5319  021456                  ALR2TS:
5320                          ;       ALTERNATE REGISTER SET R2 BIT TESTS
5321  021456  012702  177777          MOV     #177777,R2              ;R2=177777
5322  021462  020227  177777          CMP     R2,#177777              ;DOES R2=177777
5323  021466  001401                  BEQ     1$                      ;YES GO ON
5324  021470  104001                  ERROR   +1              ;CPU ERROR
5325                                                          ;NO GO TO ERROR
5326  021472  005002  1$:     CLR     R2                      ;R2=0
5327  021474  020227  000000          CMP     R2,#0                   ;DOES R2=0
5328  021500  001401                  BEQ     2$                      ;YES GO ON
5329  021502  104001                  ERROR   +1              ;CPU ERROR
5330                                                          ;NO GO TO ERROR
5331  021504  012702  125252  2$:     MOV     #125252,R2              ;R2=125252
5332  021510  020227  125252          CMP     R2,#125252              ;DOES R2=125252
5333  021514  001401                  BEQ     3$                      ;YES GO ON
5334  021516  104001                  ERROR   +1              ;CPU ERROR
5335                                                          ;NO GO TO ERROR
```

```
COKDAA0 KDJ11-B CLUSTER MACY11 30(1046)  23-JAN-84  18:56  PAGE 99
COKDAA.P11    23-JAN-84 18:55          ************ DOUBLE OPERAND TESTS ****************                SEQ 0099

 5336  021520  012702  0525ڃ3        3$:     MOV     #52525,R2               ;R2=52525
 5337  021524  020227  052525              CMP     R2,#52525               ;DOES R2=52525
 5338  021530  001401                      BEQ     4$                      ;YES GO ON
 5339  021532  104001                      ERROR   .1              ;CPU ERROR
 5340                                                                      ;NO GO TO ERROR
 5341  021534                          4$:
 5342                                  ;
 5343  021534                          ALR3TS:
 5344                                  ;       ALTERNATE REGISTER SET R3 BIT TESTS
 5345  021534  012703  177777                MOV     #177777,R3              ;R3=177777
 5346  021540  020327  177777                CMP     R3,#177777              ;DOES R3=177777
 5347  021544  001401                      BEQ     1$                      ;YES GO ON
 5348  021546  104001                      ERROR   .1              ;CPU ERROR
 5349                                                                      ;NO GO TO ERROR
 5350  021550  005003                  1$:     CLR     R3                      ;R3=0
 5351  021552  020327  000000              CMP     R3,#0                   ;DOES R3=0
 5352  021556  001401                      BEQ     2$                      ;YES GO ON
 5353  021560  104001                      ERROR   .1              ;CPU ERROR
 5354                                                                      ;NO GO TO ERROR
 5355  021562  012703  125252          2$:     MOV     #125252,R3              ;R3=125252
 5356  021566  020327  125252                CMP     R3,#125252              ;DOES R3=125252
 5357  021572  001401                      BEQ     3$                      ;YES GO ON
 5358  021574  104001                      ERROR   .1              ;CPU ERROR
 5359                                                                      ;NO GO TO ERROR
 5360  021576  012703  052525          3$:     MOV     #52525,R3               ;R3=52525
 5361  021602  020327  052525                CMP     R3,#52525               ;DOES R3=52525
 5362  021606  001401                      BEQ     4$                      ;YES GO ON
 5363  021610  104001                      ERROR   .1              ;CPU ERROR
 5364                                                                      ;NO GO TO ERROR
 5365  021612                          4$:
 5366                                  ;
 5367  021612                          ALR4TS:
 5368                                  ;       ALTERNATE REGISTER SET R4 BIT TESTS
 5369  021612  012704  177777                MOV     #177777,R4              ;R4=177777
 5370  021616  020427  177777                CMP     R4,#177777              ;DOES R4=177777
 5371  021622  001401                      BEQ     1$                      ;YES GO ON
 5372  021624  104001                      ERROR   .1              ;CPU ERROR
 5373                                                                      ;NO GO TO ERROR
 5374  021626  005004                  1$:     CLR     R4                      ;R4=0
 5375  021630  020427  000000              CMP     R4,#0                   ;DOES R4=0
 5376  021634  001401                      BEQ     2$                      ;YES GO ON
 5377  021636  104001                      ERROR   .1              ;CPU ERROR
 5378                                                                      ;NO GO TO ERROR
 5379  021640  012704  125252          2$:     MOV     #125252,R4              ;R4=125252
 5380  021644  020427  125252                CMP     R4,#125252              ;DOES R4=125252
 5381  021650  001401                      BEQ     3$                      ;YES GO ON
 5382  021652  104001                      ERROR   .1              ;CPU ERROR
 5383                                                                              ;NO GO TO ERROR
 5384  021654  012704  052525          3$:     MOV     #52525,R4               ;R4=52525
 5385  021660  020427  052525                CMP     R4,#52525               ;DOES R4=52525
 5386  021664  001401                      BEQ     4$                      ;YES GO ON
 5387  021666  104001                      ERROR   .1              ;CPU ERROR
 5388                                                                              ;NO GO TO ERROR
 5389  021670                          4$:
 5390                                  ;
 5391  021670                          ALR5TS:
```

```
                                        ;********** DOUBLE OPERAND TESTS **********                                SEQ 0100

                                        ;       ALTERNATE REGISTER SET R5 BIT TESTS
021670  012705  177777                  ;       MOV     #177777,R5              ;R5=177777
021674  020527  177777                  ;       CMP     R5,#177777              ;DOES R5=177777
021700  001401                          ;       BEQ     1$                      ;YES GO ON
021702  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
021704  005005                          ;1$:    CLR     R5                      ;R5=0
021706  020527  000000                  ;       CMP     R5,#0                   ;DOES R5=0
021712  001401                          ;       BEQ     2$                      ;YES GO ON
021714  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
021716  012705  125252                  ;2$:    MOV     #125252,R5              ;R5=125252
021722  020527  125252                  ;       CMP     R5,#125252              ;DOES R5=125252
021726  001401                          ;       BEQ     3$                      ;YES GO ON
021730  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
021732  012705  052525                  ;3$:    MOV     #52525,R5               ;R5=52525
021736  020527  052525                  ;       CMP     R5,#52525               ;DOES R5=52525
021742  001401                          ;       BEQ     4$                      ;YES GO ON
021744  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
021746  042767  004000  156022          ;4$:    BIC     #BIT11,PS               ;RETURN TO PRIMARY GEN PURPOSE REGS
                                        ;
021754                                  ;TE115:
                                        ;       TEST MOVE FROM PROCCESSOR STATUS INST (MFPS)
021754  005004                          ;       CLR     R4                      ;SETUP DESTINATION R4
                                        ;
021756  012701  022164                  ;       MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
021762  012702  022174                  ;       MOV     #TE115B,R2              ;
021766  012703  022202                  ;       MOV     #TE115C,R3              ;
021772  012137  177776          1$:     ;       MOV     (R1)+,@#177776          ;SETUP PSW
021776  106704                          ;       MFPS    R4                      ; TEST INSTRUCTION
022000  023722  177776                  ;       CMP     @#177776,(R2)+          ;CHECK PSW
022004  001401                          ;       BEQ     2$                      ;OK GO ON
022006  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
022010  020423                  2$:     ;       CMP     R4,(R3)+                ;CHECK R4
022012  001401                          ;       BEQ     3$                      ;OK GO ON
022014  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
022016  021127  177777          3$:     ;       CMP     (R1),#177777            ;ARE WE DONE
022022  001363                          ;       BNE     1$                      ;NO GO TO 1$
                                        ;
                                        ;
022024  012701  022164                  ;       MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
022030  012702  022174                  ;       MOV     #TE115B,R2              ;
022034  012703  022202                  ;       MOV     #TE115C,R3              ;
022040  010605                          ;       MOV     R6,R5                   ;SAVE STACK IN R5
022042  011137  177776          101$:   ;       MOV     (R1),@#177776           ;SETUP PSW
022046  106706                          ;       MFPS    R6                      ; TEST INSTRUCTION
022050  023712  177776                  ;       CMP     @#177776,(R2)           ;CHECK PSW
022054  001401                          ;       BEQ     102$                    ;OK GO ON
022056  104001                          ;       ERROR   .1              ;CPU ERROR
                                        ;                                       ;NO GO TO ERROR
022060  020613                  102$:   ;       CMP     R6,(R3)                 ;CHECK R6
```

```
5448  022062  001401                  BEQ     103$                    ;OK GO ON
5449  022064  104001                  ERROR   .1                      ;CPU ERROR
5450                                                                  ;NO GO TO ERROR
5451  022066  010506          103$:   MOV     R5,R6                   ;RESTORE STACK
5452
5453
5454  022070  012701  022164          MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
5455  022074  012702  022174          MOV     #TE115B,R2              ;
5456  022100  012703  022210          MOV     #TE115D,R3              ;
5457  022104  005037  110150          CLR     @#EXPDAT                ;INIT EXPECTED DATA HOLDER.
5458  022110  012704  110150    4$:   MOV     #EXPDAT,R4              ;SETUP POINTER TO TEST LOCATION
5459  022114  012137  177776          MOV     (R1)+,@#177776          ;SETUP PSW
5460  022120  106724                  MFPS    (R4)+                   ; TEST INSTRUCTION
5461  022122  023722  177776          CMP     @#177776,(R2)+          ;CHECK PSW
5462  022126  001401                  BEQ     5$                      ;OK GO ON
5463  022130  104001                  ERROR   .1                      ;CPU ERROR
5464                                                                  ;NO GO TO ERROR
5465  022132  020427  110151    5$:   CMP     R4,#EXPDAT+1            ;CHECK R4
5466  022136  001401                  BEQ     6$                      ;OK GO ON
5467  022140  104001                  ERROR   .1                      ;CPU ERROR
5468                                                                  ;NO GO TO ERROR
5469  022142  023723  110150    6$:   CMP     @#EXPDAT,(R3)+          ;CHECK TEST LOCATION
5470  022146  001401                  BEQ     7$                      ;OK GO ON
5471  022150  104001                  ERROR   .1                      ;CPU ERROR
5472                                                                  ;NO GO TO ERROR
5473  022152  021127  177777    7$:   CMP     (R1),#177777            ;ARE WE DONE
5474  022156  001354                  BNE     4$                      ;NO GO TO 4$
5475
5476  022160  000167  000032          JMP     TE115F
5477
5478                            ;
5479                            ;
5480  022164  030207            TE115A: .WORD   30207
5481  022166  030000                    .WORD   30000
5482  022170  030057                    .WORD   30057
5483  022172  177777                    .WORD   177777
5484  022174  030211            TE115B: .WORD   30211
5485  022176  030004                    .WORD   30004
5486  022200  030041                    .WORD   30041
5487  022202  177607            TE115C: .WORD   177607
5488  022204  000000                    .WORD   0
5489  022206  000057                    .WORD   57
5490  022210  000207            TE115D: .WORD   207
5491  022212  000000                    .WORD   0
5492  022214  000057                    .WORD   57
5493  022216  000240            TE115F: NOP
5494                            ;
5495  022220                    TE116:
5496                            ;       TEST MOVE TO PROCESSOR STATUS INST (MTPS)
5497
5498  022220  012737  030000  177776          MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5499  022226  012701  022522                  MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5500  022232  012702  022500                  MOV     #TE116B,R2              ;
5501  022236  010103                          MOV     R1,R3                   ;
5502  022240  004767  000136                  JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5503                                                                          ;AND SOURCE OPERAND
```

```
5504
5505
5506  022244  012737  140000  177776          MOV     #140000,@#177776        ;SET PSW TO USER MODE
5507  022252  012706  000600                  MOV     #600,R6                 ;SETUP USER STACK
5508  022256  012701  022522                  MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5509  022262  012702  022512                  MOV     #TE116C,R2              ;
5510  022266  010103                          MOV     R1,R3                   ;
5511  022270  004767  000106                  JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5512                                                                          ;AND SOURCE OPERAND
5513
5514
5515  022274  012737  140000  177776          MOV     #140000,@#177776        ;SET PSW TO USER MODE AND CLEAR CC BITS
5516  022302  012701  022474                  MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5517  022306  012702  022512                  MOV     #TE116C,R2              ;
5518  022312  012703  022522                  MOV     #TE116D,R3              ;
5519  022316  010104                          MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5520  022320  004767  000110                  JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5521                                                                          ;AND SOURCE OPERAND
5522
5523
5524  022324  012737  030000  177776          MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5525  022332  012701  022474                  MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5526  022336  012702  022500                  MOV     #TE116B,R2              ;
5527  022342  012703  022522                  MOV     #TE116D,R3              ;
5528  022346  010104                          MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5529  022350  004767  000060                  JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5530                                                                          ;AND SOURCE OPERAND
5531
5532  022354  005037  177776                  CLR     @#177776                ;SET PSW TO KERNEL MODE
5533  022360  106427  177412                  MTPS    #177412                 ;TEST INSTRUCTION
5534  022364  022737  000012  177776          CMP     #12,@#177776            ;IS PSW CORRECT
5535  022372  001401                          BEQ     100$                    ;YES GO ON
5536  022374  104001                          ERROR   +1                      ;CPU ERROR
5537                                                                          ;NO GO TO ERROR
5538  022376                          100$:
5539  022376  000167  000130                  JMP     FIN116
5540
5541  022402  012105                  T116:   MOV     (R1)+,R5                        ;MOVE TEST DATA TO R5
5542  022404  106405                          MTPS    R5                              ; TEST INSTRUCTION
5543  022406  023722  177776                  CMP     @#177776,(R2)+                  ;IS PSW CORRECT
5544  022412  001401                          BEQ     1$                              ;YES GO ON
5545  022414  104001                          ERROR   +1                      ;CPU ERROR
5546                                                                          ;NO GO TO ERROR
5547  022416  022305                  1$:     CMP     (R3)+,R5                        ;IS R5 CORRECT
5548  022420  001401                          BEQ     2$                              ;YES GO ON
5549  022422  104001                          ERROR   +1                      ;CPU ERROR
5550                                                                          ;NO GO TO ERROR
5551  022424  021227  177777          2$:     CMP     (R2),#177777            ;ARE WE DONE
5552  022430  001364                          BNE     T116                    ;NO GO TO T116
5553  022432  000207                          RTS     PC                      ;RETURN
5554
5555  022434  106421                  TA116:  MTPS    (R1)+                           ; TEST INSTRUCTION
5556  022436  023722  177776                  CMP     @#177776,(R2)+                  ;IS PSW CORRECT
5557  022442  001401                          BEQ     1$                              ;YES GO ON
5558  022444  104001                          ERROR   +1                      ;CPU ERROR
5559                                                                          ;NO GO TO ERROR
```

```
                        ************ DOUBLE OPERAND TESTS ******************                                SEQ 0103

 5560  022446  122423                1$:     CMPB    (R4)+,(R3)+             ;CHECK TEST LOCATION
 5561  022450  001401                        BEQ     2$                      ;OK GO ON
 5562  022452  104001                        ERROR   +1              ;CPU ERROR
 5563                                                                ;NO GO TO ERROR
 5564  022454  020104                2$:     CMP     R1,R4                   ;IS SOURCE OPERAND CORRECT
 5565  022456  001401                        BEQ     3$                      ;YES GO ON
 5566  022460  104001                        ERROR   +1              ;CPU ERROR
 5567                                                                ;NO GO TO ERROR
 5568  022462  005203                3$:     INC     R3                      ;POINT TO NEXT WORD
 5569  022464  021227  177777              CMP     (R2),#177777            ;ARE WE DONE
 5570  022470  001361                        BNE     TA116                   ;NO GO TO TA116
 5571  022472  000207                        RTS     PC                      ;RETURN
 5572                                ;
 5573  022474     377                TE116A: .BYTE   377
 5574  022475     000                        .BYTE   0
 5575  022476     252                        .BYTE   252
 5576  022477     125                        .BYTE   125
 5577  022500  030357                TE116B: .WORD   30357
 5578  022502  030000                        .WORD   30000
 5579  022504  030252                        .WORD   30252
 5580  022506  030105                        .WORD   30105
 5581  022510  177777                        .WORD   177777
 5582  022512  140017                TE116C: .WORD   140017
 5583  022514  140000                        .WORD   140000
 5584  022516  140012                        .WORD   140012
 5585  022520  140005                        .WORD   140005
 5586  022522  177777                TE116D: .WORD   177777
 5587  022524  177400                        .WORD   177400
 5588  022526  177652                        .WORD   177652
 5589  022530  177525                        .WORD   177525
 5590                                ;
 5591  022532                        FIN116:
 5592                                ;
 5593  022532                        TE117:
 5594                                ;       TEST MFPT (MOVE FROM PROCESSOR TYPE)
 5595  022532  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTOR
 5596  022536  012737  022644  000010        MOV     #TE117A,@#10            ;SETUP VECTOR TO HANDLE POSSIBLE ILLEGAL INST TR
 5597  022544  012700  177777                MOV     #177777,R0              ;INIT R0
 5598  022550  012737  030000  177776        MOV     #30000,@#177776         ;SETUP PSW
 5599  022556  000007                        .WORD   7                       ; TEST INSTRUCTION
 5600  022560  022737  030000  177776        CMP     #30000,@#177776         ;IS PSW CORRECT
 5601  022566  001401                        BEQ     1$                      ;YES GO ON
 5602  022570  104001                        ERROR   +1              ;CPU ERROR
 5603                                                                ;NO GO TO ERROR
 5604  022572  020027  000005          1$:   CMP     R0,#5                   ;IS R0 CORRECT
 5605  022576  001401                        BEQ     2$                      ;YES GO ON
 5606  022600  104001                        ERROR   +1              ;CPU ERROR
 5607                                                                ;NO GO TO ERROR
 5608  022602  012700  177777          2$:   MOV     #177777,R0              ;INIT R0
 5609  022606  000277                        SCC                             ;SET ALL CC BITS
 5610  022610  000007                        .WORD   7                       ; TEST INSTRUCTION
 5611  022612  022737  030017  177776        CMP     #30017,@#177776         ;IS PSW CORRECT
 5612  022620  001401                        BEQ     3$                      ;YES GO ON
 5613  022622  104001                        ERROR   +1              ;CPU ERROR
 5614                                                                ;NO GO TO ERROR
 5615  022624  020027  000005          3$:   CMP     R0,#5                   ;IS R0 CORRECT
```

```
                        ************ DOUBLE OPERAND TESTS ******************                                SEQ 0104

 5616  022630  001401                        BEQ     4$                      ;YES GO ON
 5617  022632  104001                        ERROR   .+1                  ;CPU ERROR
 5618                                                                      ;NO GO TO ERROR
 5619  022634  012637  000010        4$:     MOV     (SP)+,@#10              ;RESTORE VECTOR
 5620
 5621  022640  000167  000002                JMP     FIN117
 5622                                ;
 5623  022644  104001                TE117A: ERROR   .+1                  ;CPU ERROR
 5624                                                                      ;GO TO ERROR IF TRAP TAKES PLACE
 5625                                ;
 5626  022646  000240                FIN117: NOP
 5627                                ;
 5628  022650                        TE120:
 5629                                ;       TEST HALT (NOT KERNEL MODE)
 5630  022650  005037  177766                CLR     @#177766                ;INIT CPU ERROR REG
 5631  022654  005037  177776                CLR     @#177776                ;INIT PSW-SET KERNEL MODE
 5632  022660  013746  000004                MOV     @#4,-(SP)               ;SAVE VECTOR
 5633  022664  013746  000006                MOV     @#6,-(SP)               ;SAVE VECTOR
 5634  022670  012737  022724  000004        MOV     #TE120A,@#4             ;SET UP VECTOR TO HANDLE ILLEGAL HALT
 5635  022676  005037  000006                CLR     @#6                     ;SET UP VECTOR TO COME BACK IN KERNEL MODE
 5636  022702  012767  140000  155066        MOV     #140000,PS              ;SET IN USER MODE
 5637  022710  012706  000600                MOV     #600,R6                 ;INITIALIZE THE USER STACK POINTER
 5638  022714  000000                        HALT                            ; TEST INSTRUCTION
 5639  022716  104001                PROCNT: ERROR   .+1                  ;CPU ERROR
 5640                                                                      ;IF NOTHING HAPPENED GO TO ERROR
 5641  022720  000167  000044                JMP     FIN120                  ;
 5642                                ;
 5643                                ;
 5644  022724  022737  030000  177776 TE120A: CMP     #30000,@#177776         ;IS PSW CORRECT/PREVIOUS MODE = USER?
 5645  022732  001401                        BEQ     1$                      ;YES GO ON
 5646  022734  104001                        ERROR   .+1                  ;CPU ERROR
 5647                                                                      ;NO GO TO ERROR
 5648  022736  022737  000200  177766 1$:     CMP     #200,@#177766           ; TEST CPU ERROR REGISTER
 5649  022744  001401                        BEQ     2$                      ;YES GO ON
 5650  022746  104001                        ERROR   .+1                  ;CPU ERROR
 5651                                                                      ;NO GO TO ERROR
 5652  022750  022627  022716        2$:     CMP     (SP)+,#PROCNT           ;DOES STACK CONTAIN CORRECT PC
 5653  022754  001401                        BEQ     3$                      ;YES GO ON
 5654  022756  104001                        ERROR   .+1                  ;CPU ERROR
 5655                                                                      ;NO GO TO ERROR
 5656  022760  022627  140000        3$:     CMP     (SP)+,#140000           ;DOES STACK CONTAIN CORRECT PSW
 5657  022764  001401                        BEQ     FIN120                  ;YES GO ON
 5658  022766  104001                        ERROR   .+1                  ;CPU ERROR
 5659                                                                      ;NO GO TO ERROR
 5660  022770  012637  000006        FIN120: MOV     (SP)+,@#6               ;RESTORE VECTOR
 5661  022774  012637  000004                MOV     (SP)+,@#4               ;RESTORE VECTOR
 5662
 5663                                ;
 5664  023000                        TE121:
 5665                                ;       TEST RESET
 5666  023000  122767  000001  156212        CMPB    #APTENV,$ENV            ;ARE WE IN APT MODE?
 5667  023006  001002                        BNE     1$                      ;IF NOT: DO THIS TEST
 5668  023010  000167  000622                JMP     FIN122                  ;ELSE SKIP THIS TEST BECAUSE RESETS
 5669                                                                        ;SCREW UP THE APT MONITOR.
 5670  023014  012737  030340  177776 1$:     MOV     #30340,@#177776         ;SETUP PSW TO KERNEL MODE
 5671  023022  012737  160000  177572        MOV     #160000,@#177572        ;SETUP MMR0
```

```
5672  023030  012737  000077  172516          MOV     #77,@#172516            ;SETUP MMR3
5673  023036  005037  177772                  CLR     @#177772                ;CLEAR PIRQ
5674  023042  023727  177772  000000          CMP     @#177772,#0             ;IS PIRQ CORRECT
5675  023050  001401                          BEQ     C121A                   ;YES GO ON
5676  023052  104001                          ERROR   ;1              ;CPU ERROR
5677                                                                  ;NO GO TO ERROR
5678  023054  012737  025000  177772  C121A:  MOV     #25000,@#177772         ;MOVE AN ALTERNATING PATTERN TO PIRQ
5679  023062  022737  025252  177772          CMP     #25252,@#177772         ;IS PIRQ CORRECT
5680  023070  001401                          BEQ     C121B                   ;YES GO ON
5681  023072  104001                          ERROR   ;1              ;CPU ERROR
5682                                                                  ;NO GO TO ERROR
5683  023074  012737  077000  177772  C121B:  MOV     #77000,@#177772         ;SETUP PIRQ
5684  023102  022737  077314  177772          CMP     #77314,@#177772         ;IS PIRQ CORRECT
5685  023110  001401                          BEQ     C121C                   ;YES GO ON
5686  023112  104001                          ERROR   ;1              ;CPU ERROR
5687                                                                  ;NO GO TO ERROR
5688  023114  000277                  C121C:  SCC                             ;SET ALL CC BITS
5689  023116  000005                          RESET                           ; TEST INSTRUCTION
5690  023120  022737  030357  177776          CMP     #30357,@#177776         ;IS PSW CORRECT
5691  023126  001401                          BEQ     1$                      ;YES GO ON
5692  023130  104001                          ERROR   ;1              ;CPU ERROR
5693                                                                  ;NO GO TO ERROR
5694  023132  013701  177572          1$:     MOV     @#SR0,R1                ;SAVE SR0 IN R1.
5695  023136  042701  000176                  BIC     #1 6,R1                 ;STRIP OFF UNDEFINED BITS 1-6 FROM MMR0
5696  023142  022701  000000                  CMP     #0,R1                   ;IS MMR0 CORRECT
5697  023146  001401                          BEQ     2$                      ;YES GO ON
5698  023150  104001                          ERROR   ;1              ;CPU ERROR
5699                                                                  ;NO GO TO ERROR
5700  023152  022737  000000  172516  2$:     CMP     #0,@#172516             ;IS MMR3 CORRECT
5701  023160  001401                          BEQ     3$                      ;YES GO ON
5702  023162  104001                          ERROR   ;1              ;CPU ERROR
5703                                                                  ;NO GO TO ERROR
5704  023164  022737  000000  177772  3$:     CMP     #0,@#177772             ;IS PIRQ CORRECT
5705  023172  001401                          BEQ     4$                      ;YES GO ON
5706  023174  104001                          ERROR   ;1              ;CPU ERROR
5707                                                                  ;NO GO TO ERROR
5708
5709  023176  013702  000004          4$:     MOV     @#4,R2                  ;SAVE LOC 4 IN R2
5710  023202  013703  000006                  MOV     @#6,R3                  ;SAVE LOC 6 IN R3
5711  023206  012737  023344  000004          MOV     #4$,@#4                 ;SETUP VECTORS IN CASE AN ERROR CAUSES
5712                                                                          ;A HALT TO OCCUR IN USER MODE.
5713  023214  012737  000340  000006          MOV     #340,@#6                ;THIS SETS KERNEL MODE, AND PREVENTS PRIQ
5714                                                                          ;INTERRUPTS FROM TRASHING THE STACK IF A
5715                                                                          ;USER MODE HALT OCCURS.
5716  023222  012737  140340  177776          MOV     #140340,@#177776        ;SETUP PSW TO USER MODE
5717  023230  012737  160000  177572          MOV     #160000,@#177572        ;SETUP MMR0
5718  023236  012737  000077  172516          MOV     #77,@#172516            ;SETUP MMR3
5719  023244  012737  077000  177772          MOV     #77000,@#177772         ;SETUP PIRQ
5720  023252  000277                          SCC                             ;SET ALL CC BITS
5721  023254  000005                          RESET                           ; TEST INSTRUCTION
5722  023256  022737  140357  177776          CMP     #140357,@#177776        ;IS PSW CORRECT
5723  023264  001402                          BEQ     5$                      ;YES GO ON
5724  023266  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
5725  023270  104001                          ERROR   ;1              ;CPU ERROR
5726                                                                  ;NO GO TO ERROR
5727  023272  013701  177572          5$:     MOV     @#177572,R1             ;SAVE MMR0 IN R1
```

```
5728  023276  042701  000176                  BIC     #176,R1                 ;CLEAR BITS 1-6 FROM MMR0
5729  023302  022701  160000                  CMP     #160000,R1              ;IS MMR0 CORRECT
5730  023306  001402                          BEQ     6$                      ;YES GO ON
5731  023310  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
5732  023312  104001                          ERROR   .1              ;CPU ERROR
5733                                                                  ;NO GO TO ERROR
5734  023314  022737  000077  172516  6$:     CMP     #77,@#172516            ;IS MMR3 CORRECT
5735  023322  001402                          BEQ     7$                      ;YES GO ON
5736  023324  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
5737  023326  104001                          ERROR   .1              ;CPU ERROR
5738                                                                  ;NO GO TO ERROR
5739  023330  022737  077314  177772  7$:     CMP     #77314,@#177772         ;IS PIRQ CORRECT
5740  023336  001403                          BEQ     9$                      ;YES GO ON
5741  023340  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
5742  023342  104001                           ERROR   .1              ;CPU ERROR
5743                                                                  ;NO GO TO ERROR
5744  023344  000002                  8$:     RTI                             ;USER MODE HALT OCCURRED; GO TO ERROR.
5745
5746  023346  005037  177772          9$:     CLR     @#177772                ;CLEAR PIRQ
5747  023352  005037  177776                  CLR     @#177776                ;CLEAR PSW
5748  023356  010237  000004                  MOV     R2,@#4                  ;RESTORE VECTORS TO PREVIOUS STATE
5749  023362  010337  000006                  MOV     R3,@#6                  ;
5750  023366                          FIN121:
5751                                  ;
5752  023366                          TE122:
5753                                  ;       TEST SPL (SET PRIORITY LEVEL)
5754
5755  023366  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
5756  023372  012701  023616                  MOV     #T122B,R1               ;SETUP POINTER TO DATA
5757  023376  012737  030000  177776  1$:     MOV     #30000,@#177776         ;INIT PSW
5758  023404  004767  000054                  JSR     PC,T122A                ; TEST INSTRUCTION
5759  023410  022137  177776                  CMP     (R1)+,@#177776          ;IS PSW CORRECT
5760  023414  001401                          BEQ     2$                      ;YES GO ON
5761  023416  104001                          ERROR   .1              ;CPU ERROR
5762                                                                  ;NO GO TO ERROR
5763  023420  077512                  2$:     SOB     R5,1$                   ;REPEAT UNTIL ALL CASES ARE TESTED
5764
5765
5766  023422  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
5767  023426  012737  140000  177776  3$:     MOV     #140000,@#177776        ;SETUP PSW TO USER MODE
5768  023434  012706  000600                  MOV     #600,R6                 ;SETUP USER STACK
5769  023440  004767  000020                  JSR     PC,T122A                ; TEST INSTRUCTION
5770  023444  022737  140017  177776          CMP     #140017,@#177776        ;IS PSW CORRECT
5771  023452  001401                          BEQ     4$                      ;YES GO ON
5772  023454  104001                          ERROR   .1              ;CPU ERROR
5773                                                                  ;NO GO TO ERROR
5774  023456  077515                  4$:     SOB     R5,3$                   ;REPEAT UNTIL ALL CASES ARE TESTED
5775
5776
5777  023460  000167  000152                  JMP     FIN122
5778                                  ;
5779  023464  020527  000010          T122A:  CMP     R5,#8.                  ;FIND OUT WHAT COUNTER IS
5780  023470  001003                          BNE     1$                      ;IF NOT PRIORITY 0 GO TO 1$
5781  023472  000277                          SCC                             ;SET ALL CC BITS
5782  023474  000230                          SPL     0                       ;SET PRIORITY TO 0
5783  023476  000446                          BR      8$                      ;RETURN
```

```
5784  023500  020527  000007          1$:     CMP     R5,#7                   ;FIND OUT WHAT COUNTER IS
5785  023504  001003                          BNE     2$                      ;IF NOT PRIORITY 1 GO TO 2$
5786  023506  000277                          SCC                             ;SET ALL CC BITS
5787  023510  000231                          SPL     1                       ;SET PRIORITY TO 1
5788  023512  000440                          BR      8$                      ;RETURN
5789  023514  020527  000006          2$:     CMP     R5,#6                   ;FIND OUT WHAT COUNTER IS
5790  023520  001003                          BNE     3$                      ;IF NOT PRIORITY 2 GO TO 3$
5791  023522  000277                          SCC                             ;SET ALL CC BITS
5792  023524  000232                          SPL     2                       ;SET PRIORITY TO 2
5793  023526  000432                          BR      8$                      ;RETURN
5794  023530  020527  000005          3$:     CMP     R5,#5                   ;FIND OUT WHAT COUNTER IS
5795  023534  001003                          BNE     4$                      ;IF NOT PRIORITY 3 GO TO 4$
5796  023536  000277                          SCC                             ;SET ALL CC BITS
5797  023540  000233                          SPL     3                       ;SET PRIORITY TO 3
5798  023542  000424                          BR      8$                      ;RETURN
5799  023544  020527  000004          4$:     CMP     R5,#4                   ;FIND OUT WHAT COUNTER IS
5800  023550  001003                          BNE     5$                      ;IF NOT PRIORITY 4 GO TO 5$
5801  023552  000277                          SCC                             ;SET ALL CC BITS
5802  023554  000234                          SPL     4                       ;SET PRIORITY TO 4
5803  023556  000416                          BR      8$                      ;RETURN
5804  023560  020527  000003          5$:     CMP     R5,#3                   ;FIND OUT WHAT COUNTER IS
5805  023564  001003                          BNE     6$                      ;IF NOT PRIORITY 5 GO TO 6$
5806  023566  000277                          SCC                             ;SET ALL CC BITS
5807  023570  000235                          SPL     5                       ;SET PRIORITY TO 5
5808  023572  000410                          BR      8$                      ;RETURN
5809  023574  020527  000002          6$:     CMP     R5,#2                   ;FIND OUT WHAT COUNTER IS
5810  023600  001003                          BNE     7$                      ;IF NOT PRIORITY 6 GO TO 7$
5811  023602  000277                          SCC                             ;SET ALL CC BITS
5812  023604  000236                          SPL     6                       ;SET PRIORITY TO 6
5813  023606  000402                          BR      8$                      ;RETURN
5814  023610  000277                  7$:     SCC                             ;SET ALL CC BITS
5815  023612  000237                          SPL     7                       ;SET PRIORITY TO 7
5816  023614  000207                  8$:     RTS     PC                      ;RETURN
5817                                  ;
5818  023616  030017                  T1228:  .WORD   30017
5819  023620  030057                          .WORD   30057
5820  023622  030117                          .WORD   30117
5821  023624  030157                          .WORD   30157
5822  023626  030217                          .WORD   30217
5823  023630  030257                          .WORD   30257
5824  023632  030317                          .WORD   30317
5825  023634  030357                          .WORD   30357
5826  023636  000240                  FIN122: NOP
5827                                  ;
5828  023640                          TE123:
5829                                  ;       TEST TSTSET INSTRUCTION (MULTI PROCESSING INST)
5830  023640  005037  177776                  CLR     @#177776                ;INIT PSW
5831  023644  012703  000012                  MOV     #10.,R3                 ;INIT COUNTER
5832  023650  012701  000400                  MOV     #400,R1                 ;SETUP DESTINATION
5833  023654  012700  024022                  MOV     #T123A,R0               ;SETUP SOURCE
5834  023660  012021                  100$:   MOV     (R0)+,(R1)+             ;RELOCATE TABLES
5835  023662  077302                          SOB     R3,100$                 ;ARE WE DONE
5836  023664  013746  000010                  MOV     @#10,-(SP)              ;SAVE VECTOR
5837  023670  012737  024046  000010          MOV     #T123D,@#10             ;SETUP NEW VECTOR
5838  023676  005000                          CLR     R0                      ;INIT R0
5839  023700  012701  000400                  MOV     #400,R1                 ;SETUP POINTERS TO TABLES
```

```
 5840  023704  012702  000410                 MOV     #410,R2                 ;
 5841  023710  012703  000416                 MOV     #416,R3                 ;
 5842  023714  010104                         MOV     R1,R4                   ;
 5843  023716  012737  030000  177776  1$:    MOV     #30000,@#177776         ;SETUP PSW
 5844  023724  000262                         SEV                             ;SET V BIT
 5845  023726  007221                         .WORD   7221                    ; TEST INSTRUCTION
 5846  023730  022237  177776                 CMP     (R2)+,@#177776          ;IS PSW CORRECT
 5847  023734  001401                         BEQ     2$                      ;YES GO ON
 5848  023736  104001                         ERROR   .1              ;CPU ERROR
 5849                                                                 ;NO GO TO ERROR
 5850  023740  020013                 2$:     CMP     R0,(R3)                 ;IS R0 CORRECT
 5851  023742  001401                         BEQ     3$                      ;YES GO ON
 5852  023744  104001                         ERROR   .1              ;CPU ERROR
 5853                                                                 ;NO GO TO ERROR
 5854  023746  005204                 3$:     INC     R4                      ;SETUP EXPECTED DATA
 5855  023750  005204                         INC     R4                      ;
 5856  023752  020401                         CMP     R4,R1                   ;IS R1 CORRECT
 5857  023754  001401                         BEQ     4$                      ;YES GO ON
 5858  023756  104001                         ERROR   .1              ;CPU ERROR
 5859                                                                 ;NO GO TO ERROR
 5860  023760  052713  000001         4$:     BIS     #1,(R3)                 ;SETUP EXPECTED DATA
 5861  023764  022341                         CMP     (R3)+,-(R1)             ;IS TEST LOCATION CORRECT
 5862  023766  001401                         BEQ     5$                      ;YES GO ON
 5863  023770  104001                         ERROR   .1              ;CPU ERROR
 5864                                                                 ;NO GO TO ERROR
 5865  023772  005201                 5$:     INC     R1                      ;POINT TO NEXT TEST LOCATION
 5866  023774  005201                         INC     R1                      ;
 5867  023776  021127  177777                 CMP     (R1),#177777            ;ARE WE DONE
 5868  024002  001345                         BNE     1$                      ;NO GO TO 1$
 5869  024004  012737  024050  000010         MOV     #T123E,@#10             ;SETUP NEW VECTOR
 5870  024012  007201                         .WORD   7201                    ; TEST INSTRUCTION ILLEGAL MODE
 5871  024014  104001                         ERROR   .1              ;CPU ERROR
 5872                                                                 ;GO TO ERROR IF DIDN'T TRAP
 5873  024016  000167  000032                 JMP     T123F
 5874
 5875                                 ;
                                      ;
 5876  024022  167604                 T123A:  .WORD   167604
 5877  024024  000000                         .WORD   0
 5878  024026  000001                         .WORD   1
 5879  024030  177777                         .WORD   177777
 5880  024032  030010                 T123B:  .WORD   30010
 5881  024034  030004                         .WORD   30004
 5882  024036  030001                         .WORD   30001
 5883  024040  167604                 T123C:  .WORD   167604
 5884  024042  000000                         .WORD   0
 5885  024044  000001                         .WORD   1
 5886  024046  104001                 T123D:  ERROR   .1              ;CPU ERROR
 5887                                                                 ;GO TO ERROR IF TRAPPED
 5888  024050  005726                 T123E:  TST     (SP)+                   ;CLEAN UP STACK
 5889  024052  005726                         TST     (SP)+                   ;
 5890  024054  012637  000010         T123F:  MOV     (SP)+,@#10              ;RESTORE VECTOR
 5891
 5892                                 ;
 5893  024060                         TE124:
 5894                                 ;       TEST WRTLCK (WRITE LOCK MULTI PROCESSING INST)
 5895  024060  005037  177776                 CLR     @#177776                ;INIT PSW
```

```
 5896  024064  012703  000012                 MOV     #10.,R3                 ;INIT COUNTER
 5897  024070  012701  000400                 MOV     #400,R1                 ;SETUP DESTINATION
 5898  024074  012700  024254                 MOV     #T124A,R0               ;SETUP SOURCE
 5899  024100  012021                 100$:   MOV     (R0)+,(R1)+             ;RELOCATE TABLES
 5900  024102  077302                         SOB     R3,100$                 ;ARE WE DONE
 5901  024104  013746  000010                 MOV     @#10,-(SP)              ;SAVE VECTOR
 5902  024110  012737  024300  000010         MOV     #T124D,@#10             ;SETUP NEW VECTOR
 5903  024116  012701  000400                 MOV     #400,R1                 ;SETUP POINTERS TO TABLES
 5904  024122  012702  000410                 MOV     #410,R2                 ;
 5905  024126  012703  000416                 MOV     #416,R3                 ;
 5906  024132  010204                         MOV     R2,R4                   ;
 5907  024134  012737  030000  177776  1$:    MOV     #30000,@#177776         ;SETUP PSW
 5908  024142  011100                         MOV     (R1),R0                 ;SETUP R0
 5909  024144  020327  000416                 CMP     R3,#416                 ;IS THIS THE FIRST TEST CASE
 5910  024150  001401                         BEQ     2$                      ;YES GO TO 2$
 5911  024152  000402                         BR      3$                      ;NO GO TO 3$
 5912  024154  000261                 2$:     SEC                             ;SET C BIT
 5913  024156  000401                         BR      4$                      ;
 5914  024160  000241                 3$:     CLC                             ;CLEAR C BIT
 5915  024162  000262                 4$:     SEV                             ;SET V BIT
 5916  024164  007322                         .WORD   7322                    ; TEST INSTRUCTION
 5917  024166  022337  177776                 CMP     (R3)+,@#177776          ;IS PSW CORRECT
 5918  024172  001401                         BEQ     5$                      ;YES GO ON
 5919  024174  104001                         ERROR   +1              ;CPU ERROR
 5920                                                                 ;NO GO TO ERROR
 5921  024176  021100                 5$:     CMP     (R1),R0                 ;IS R0 CORRECT
 5922  024200  001401                         BEQ     6$                      ;YES GO ON
 5923  024202  104001                         ERROR   +1              ;CPU ERROR
 5924                                                                 ;NO GO TO ERROR
 5925  024204  005204                 6$:     INC     R4                      ;SETUP EXPECTED DATA
 5926  024206  005204                         INC     R4                      ;
 5927  024210  020204                         CMP     R2,R4                   ;IS R2 CORRECT
 5928  024212  001401                         BEQ     7$                      ;YES GO ON
 5929  024214  104001                         ERROR   +1              ;CPU ERROR
 5930                                                                 ;NO GO TO ERROR
 5931  024216  022142                 7$:     CMP     (R1)+,-(R2)             ;IS TEST LOCATION CORRECT
 5932  024220  001401                         BEQ     8$                      ;YES GO ON
 5933  024222  104001                         ERROR   +1              ;CPU ERROR
 5934                                                                 ;NO GO TO ERROR
 5935  024224  005202                 8$:     INC     R2                      ;POINT TO NEXT TEST LOCATION
 5936  024226  005202                         INC     R2                      ;
 5937  024230  021127  177777                 CMP     (R1),#177777            ;ARE WE DONE
 5938  024234  001337                         BNE     1$                      ;NO GO TO 1$
 5939  024236  012737  024302  000010         MOV     #T124E,@#10             ;SETUP NEW VECTOR
 5940  024244  007302                         .WORD   7302                    ; TEST INSTRUCTION ILLEGAL MODE
 5941  024246  104001                         ERROR   +1              ;CPU ERROR
 5942                                                                 ;GO TO ERROR IF DIDN'T TRAP
 5943  024250  000167  000032                 JMP     T124F
 5944                                 ;
 5945                                 ;
 5946  024254  167604                 T124A:  .WORD   167604
 5947  024256  000000                         .WORD   0
 5948  024260  000001                         .WORD   1
 5949  024262  177777                         .WORD   177777
 5950  024264  177777                 T124B:  .WORD   177777
 5951  024266  177777                         .WORD   177777
```

```
 5952  024270  177777                          .WORD   177777
 5953  024272  030011                T124C:    .WORD   30011
 5954  024274  030004                          .WORD   30004
 5955  024276  030000                          .WORD   30000
 5956  024300  104001                T124D:    ERROR   .1                ;CPU ERROR
 5957                                                                     ;GO TO ERROR IF TRAPPED
 5958  024302  005726                T124E:    TST     (SP).+                   ;CLEAN UP STACK
 5959  024304  005726                          TST     (SP).+                   ;
 5960  024306  012637  000010        T124F:    MOV     (SP).+,@#10              ;RESTORE VECTOR
 5961
 5962                                ;
 5963  024312                        TE125:
 5964                                ;       TEST MUL (MULTIPLY INST)
 5965  024312  005037  177776                  CLR     @#177776                 ;INIT PS
 5966  024316  012701  024552                  MOV     #TE125A,R1               ;SETUP POINTERS TO TABLES
 5967
 5968  024322  010137  110150        1$:       MOV     R1,@#EXPDAT              ;
 5969  024326  062737  000002  110150          ADD     #2,@#EXPDAT              ;POINT TO SOURCE
 5970  024334  012703  122222                  MOV     #122222,R3               ;INIT R3 TO A KNOWN STATE
 5971  024340  011102                          MOV     (R1),R2          ;INIT DESTINATION REG
 5972  024342  000277                          SCC                              ;SET ALL CC BITS
 5973  024344  070261  000002                  MUL     2(R1),R2                 ; TEST INSTRUCTION
 5974  024350  026137  000004  177776          CMP     4(R1),@#177776           ;IS PS CORRECT
 5975  024356  001401                          BEQ     2$                       ;YES GO ON
 5976  024360  104001                          ERROR   .1               ;CPU ERROR
 5977                                                                     ;NO GO TO ERROR
 5978  024362  026103  000006        2$:       CMP     6(R1),R3                 ;IS R3 CORRECT
 5979  024366  001401                          BEQ     3$                       ;YES GO ON
 5980  024370  104001                          ERROR   .1               ;CPU ERROR
 5981                                                                     ;NO GO TO ERROR
 5982  024372  026102  000010        3$:       CMP     10(R1),R2                ;IS R2 CORRECT
 5983  024376  001401                          BEQ     4$                       ;YES GO ON
 5984  024400  104001                          ERROR   .1               ;CPU ERROR
 5985                                                                     ;NO GO TO ERROR
 5986  024402  026177  000002  063540  4$:       CMP     2(R1),@EXPDAT            ;IS SOURCE LOCATION OK
 5987  024410  001401                          BEQ     5$                       ;YES GO ON
 5988  024412  104001                          ERROR   .1               ;CPU ERROR
 5989                                                                     ;NO GO TO ERROR
 5990  024414  062701  000012        5$:       ADD     #12,R1                   ;GO TO NEXT TEST
 5991  024420  020127  025000                  CMP     R1,#FIN125               ;ARE WE FINISHED
 5992  024424  001336                          BNE     1$                       ;NO GO TO 1$
 5993
 5994                                ;
 5995                                ;SECOND PART
 5996                                ;USING ODD REGISTER
 5997                                ;
 5998  024426  012701  024552        6$:       MOV     #TE125A,R1               ;SETUP POINTERS TO TABLES
 5999  024432                        7$:
 6000  024432  010102                          MOV     R1,R2                    ;
 6001  024434  012706  001000                  MOV     #STBOT,R6                ;INIT R6 TO A KNOWN STATE
 6002  024440  012704  000004                  MOV     #4,R4                    ;SETUP R4 VALUE
 6003  024444  011105                          MOV     (R1),R5          ;INIT DESTINATION REG
 6004  024446  000277                          SCC                              ;SET ALL CC BITS
 6005  024450  070561  000002                  MUL     2(R1),R5                 ; TEST INSTRUCTION
 6006  024454  026137  000004  177776          CMP     4(R1),@#177776           ;IS PS CORRECT
 6007  024462  001401                          BEQ     8$                       ;YES GO ON
```

```
6008  024464  104001                  ERROR   .1              ;CPU ERROR
6009                                                          ;NO GO TO ERROR
6010  024466  026105  000006    8$:   CMP     6(R1),R5        ;IS R5 CORRECT
6011  024472  001401                  BEQ     9$              ;YES GO ON
6012  024474  104001                  ERROR   .1              ;CPU ERROR
6013                                                          ;NO GO TO ERROR
6014  024476  020627  001000    9$:   CMP     R6,@STBOT       ;IS R6 CORRECT
6015  024502  001403                  BEQ     10$             ;YES GO ON
6016  024504  012706  001000        MOV     @STBOT,R6       ;RESTORE SP
6017  024510  104001                  ERROR   .1              ;CPU ERROR
6018                                                          ;NO GO TO ERROR
6019  024512  005722    10$:  TST     (R2)+           ;POINT TO SOURCE OPERAND
6020  024514  021261  000002        CMP     (R2),2(R1)      ;IS SOURCE LOCATION OK
6021  024520  001401                  BEQ     11$             ;YES GO ON
6022  024522  104001                  ERROR   .1              ;CPU ERROR
6023                                                          ;NO GO TO ERROR
6024  024524  020427  000004    11$:  CMP     R4,@4           ;IS R4 CORRECT
6025  024530  001401                  BEQ     12$             ;YES GO ON
6026  024532  104001                  ERROR   .1              ;CPU ERROR
6027                                                          ;NO GO TO ERROR
6028  024534  062701  000012    12$:  ADD     @12,R1          ;
6029  024540  020127  025000        CMP     R1,@FIN125      ;ARE WE FINISHED
6030  024544  001332                  BNE     7$              ;NO GO TO 7$
6031
6032
6033  024546  000167  000226        JMP     FIN125
6034                                ;
6035                                ;
6036  024552  177777    TE125A: .WORD   177777          ;MULTIPLICAND
6037  024554  177777            .WORD   177777          ;MULTIPLIER
6038  024556  000000            .WORD   0
6039  024560  000001            .WORD   1
6040  024562  000000            .WORD   0
6041
6042  024564  006772            .WORD   6772            ;MULTIPLICAND
6043  024566  100000            .WORD   100000          ;MULTIPLIER
6044  024570  000011            .WORD   11
6045  024572  000000            .WORD   0
6046  024574  174403            .WORD   174403
6047
6048  024576  177777            .WORD   177777          ;MULTIPLICAND
6049  024600  077777            .WORD   77777           ;MULTIPLIER
6050  024602  000010            .WORD   10
6051  024604  100001            .WORD   100001
6052  024606  177777            .WORD   177777
6053
6054  024610  077777            .WORD   77777           ;MULTIPLICAND
6055  024612  000456            .WORD   456             ;MULTIPLIER
6056  024614  000001            .WORD   1
6057  024616  177322            .WORD   177322
6058  024620  000226            .WORD   226
6059
6060  024622  173210            .WORD   173210          ;MULTIPLICAND
6061  024624  000000            .WORD   0               ;MULTIPLIER
6062  024626  000004            .WORD   4
6063  024630  000000            .WORD   0
```

```
6064  024632  000000                          .WORD   0
6065
6066  024634  000000                          .WORD   0                       ;MULTIPLICAND
6067  024636  003251                          .WORD   3251                    ;MULTIPLIER
6068  024640  000004                          .WORD   4
6069  024642  000000                          .WORD   0
6070  024644  000000                          .WORD   0
6071
6072  024646  000000                          .WORD   0                       ;MULTIPLICAND
6073  024650  000000                          .WORD   0                       ;MULTIPLIER
6074  024652  000004                          .WORD   4
6075  024654  000000                          .WORD   0
6076  024656  000000                          .WORD   0
6077
6078  024660  100000                          .WORD   100000                  ;MULTIPLICAND
6079  024662  000001                          .WORD   1                       ;MULTIPLIER
6080  024664  000010                          .WORD   10
6081  024666  100000                          .WORD   100000
6082  024670  177777                          .WORD   177777
6083
6084  024672  077777                          .WORD   77777                   ;MULTIPLICAND
6085  024674  000001                          .WORD   1                       ;MULTIPLIER
6086  024676  000000                          .WORD   0
6087  024700  077777                          .WORD   77777
6088  024702  000000                          .WORD   0
6089
6090  024704  000010                          .WORD   10                      ;MULTIPLICAND
6091  024706  010000                          .WORD   10000                   ;MULTIPLIER
6092  024710  000001                          .WORD   1
6093  024712  100000                          .WORD   100000
6094  024714  000000                          .WORD   0
6095
6096  024716  001452                          .WORD   1452                    ;MULTIPLICAND
6097  024720  034527                          .WORD   34527                   ;MULTIPLIER
6098  024722  000001                          .WORD   1
6099  024724  066506                          .WORD   66506
6100  024726  000265                          .WORD   265
6101
6102  024730  000007                          .WORD   7                       ;MULTIPLICAND
6103  024732  000400                          .WORD   400                     ;MULTIPLIER
6104  024734  000000                          .WORD   0
6105  024736  003400                          .WORD   3400
6106  024740  000000                          .WORD   0
6107
6108  024742  000002                          .WORD   2                       ;MULTIPLICAND
6109  024744  100000                          .WORD   100000                  ;MULTIPLIER
6110  024746  000011                          .WORD   11
6111  024750  000000                          .WORD   0
6112  024752  177777                          .WORD   177777
6113
6114  024754  100000                          .WORD   100000                  ;MULTIPLICAND
6115  024756  077777                          .WORD   77777                   ;MULTIPLIER
6116  024760  000011                          .WORD   11
6117  024762  100000                          .WORD   100000
6118  024764  140000                          .WORD   140000
6119
```

```
6120  024766  000001                          .WORD   1                       ;MULTIPLICAND
6121  024770  177777                          .WORD   177777                  ;MULTIPLIER
6122  024772  000010                          .WORD   10
6123  024774  177777                          .WORD   177777
6124  024776  177777                          .WORD   177777
6125  025000                          FIN125:
6126                                  ;
6127  025000                          TE126:
6128                                  ;       TEST DIV (DIVIDE INST)
6129  025000  005037  177776                  CLR     @#177776                ;INIT PSW
6130  025004  005006                          CLR     R6                      ;INIT SP
6131  025006  013705  000000                  MOV     @#0,R5                  ;SAVE VECTORS
6132  025012  013701  000002                  MOV     @#2,R1                  ;
6133  025016  012737  000137  000000          MOV     #137,@#0                ;SETUP NEW VECTORS
6134  025024  012737  025046  000002          MOV     #TE126A,@#2             ;
6135  025032  000277                          SCC                             ;SET ALL CC BITS
6136  025034  071627  000002                  DIV     #2,R6                   ; TEST INSTRUCTION
6137  025040  012706  001000          A126:   MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6138  025044  104001                          ERROR   +1                      ;CPU ERROR
6139                                                                          ;IF R7 ISN'T CORRECT GO TO ERROR
6140  025046  022737  000000  177776  TE126A: CMP     #0,@#177776             ;IS PS CORRECT
6141  025054  001403                          BEQ     1$                      ;YES GO ON
6142  025056  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6143  025062  104001                          ERROR   +1                      ;CPU ERROR
6144                                                                          ;NO GO TO ERROR
6145  025064  012704  025040          1$:     MOV     #A126,R4                ;SETUP EXPECTED DATA
6146  025070  006204                          ASR     R4                      ;
6147  025072  020406                          CMP     R4,R6                   ;IS R6 CORRECT
6148  025074  001403                          BEQ     2$                      ;YES GO ON
6149  025076  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6150  025102  104001                          ERROR   +1                      ;CPU ERROR
6151                                                                          ;NO GO TO ERROR
6152  025104  010537  000000          2$:     MOV     R5,@#0                  ;RESTORE VECTORS
6153  025110  010137  000002                  MOV     R1,@#2                  ;
6154  025114  012706  001000                  MOV     #STBOT,R6               ;INIT SP
6155  025120  012702  000006                  MOV     #6,R2                   ;INIT GPR 2
6156  025124  012703  000047                  MOV     #47,R3                  ;INIT GPR 3
6157  025130  000277                          SCC                             ;SET ALL CC BITS
6158  025132  071302                          DIV     R2,R3                   ; TEST INSTRUCTION
6159  025134  022737  000002  177776          CMP     #2,@#177776             ;IS PS CORRECT
6160  025142  001401                          BEQ     3$                      ;YES GO ON
6161  025144  104001                          ERROR   +1                      ;CPU ERROR
6162                                                                          ;NO GO TO ERROR
6163  025146  022702  000006          3$:     CMP     #6,R2                   ;IS R2 CORRECT
6164  025152  001401                          BEQ     4$                      ;YES GO ON
6165  025154  104001                          ERROR   +1                      ;CPU ERROR
6166                                                                          ;NO GO TO ERROR
6167  025156  022703  000047          4$:     CMP     #47,R3                  ;IS R3 CORRECT
6168  025162  001401                          BEQ     5$                      ;YES GO ON
6169  025164  104001                          ERROR   +1                      ;CPU ERROR
6170                                                                          ;NO GO TO ERROR
6171  025166  005004                  5$:     CLR     R4                      ;INIT R4
6172  025170  012705  000004                  MOV     #4,R5                   ;INIT R5
6173  025174  000277                          SCC                             ;SET ALL CC BITS
6174  025176  071427  000000                  DIV     #0,R4                   ; TEST INSTRUCTION
6175  025202  022737  000007  177776          CMP     #7,@#177776             ;IS PS CORRECT
```

```
 6176  025210  001401                        BEQ     6$                      ;YES GO ON
 6177  025212  104001                        ERROR   .1                  ;CPU ERROR
 6178                                                                    ;NO GO TO ERROR
 6179  025214  022704  000000          6$:   CMP     #0,R4                   ;IS R4 CORRECT
 6180  025220  001401                        BEQ     7$                      ;YES GO ON
 6181  025222  104001                        ERROR   .1                  ;CPU ERROR
 6182                                                                    ;NO GO TO ERROR
 6183  025224  022705  000004          7$:   CMP     #4,R5                   ;IS R5 CORRECT
 6184  025230  001401                        BEQ     8$                      ;YES GO ON
 6185  025232  104001                        ERROR   .1                  ;CPU ERROR
 6186                                                                    ;NO GO TO ERROR
 6187  025234  012700  000004          8$:   MOV     #4,R0                   ;INIT R0
 6188  025240  012705  000010                MOV     #10,R5                  ;INIT R5
 6189  025244  005004                        CLR     R4                      ;INIT R4
 6190  025246  000277                        SCC                             ;SET ALL CC BITS
 6191  025250  071400                        DIV     R0,R4                   ; TEST INSTRUCTION
 6192  025252  022737  000000  177776        CMP     #0,@#177776             ;IS PS CORRECT
 6193  025260  001405                        BEQ     9$                      ;YES GO ON
 6194  025262  010067  153112                MOV     R0,400                  ;SAVE R0
 6195  025266  104001                        ERROR   .1                  ;CPU ERROR
 6196                                                                    ;NO GO TO ERROR
 6197  025270  016700  153104                MOV     400,R0                  ;RESTORE R0
 6198  025274  022700  000004          9$:   CMP     #4,R0                   ;IS R0 CORRECT
 6199  025300  001401                        BEQ     10$                     ;YES GO ON
 6200  025302  104001                        ERROR   .1                  ;CPU ERROR
 6201                                                                    ;NO GO TO ERROR
 6202  025304  022704  000002          10$:  CMP     #2,R4                   ;IS R4 CORRECT
 6203  025310  001401                        BEQ     11$                     ;YES GO ON
 6204  025312  104001                        ERROR   .1                  ;CPU ERROR
 6205                                                                    ;NO GO TO ERROR
 6206  025314  022705  000000          11$:  CMP     #0,R5                   ;IS R5 CORRECT
 6207  025320  001401                        BEQ     12$                     ;YES GO ON
 6208  025322  104001                        ERROR   .1                  ;CPU ERROR
 6209                                                                    ;NO GO TO ERROR
 6210  025324  012705  000010          12$:  MOV     #10,R5                  ;INIT R5
 6211  025330  005004                        CLR     R4                      ;INIT R4
 6212  025332  000277                        SCC                             ;SET ALL CC BITS
 6213  025334  071427  000003                DIV     #3,R4                   ; TEST INSTRUCTION
 6214  025340  022737  000000  177776        CMP     #0,@#177776             ;IS PS CORRECT
 6215  025346  001401                        BEQ     13$                     ;YES GO ON
 6216  025350  104001                        ERROR   .1                  ;CPU ERROR
 6217                                                                    ;NO GO TO ERROR
 6218  025352  022704  000002          13$:  CMP     #2,R4                   ;IS R4 CORRECT
 6219  025356  001401                        BEQ     14$                     ;YES GO ON
 6220  025360  104001                        ERROR   .1                  ;CPU ERROR
 6221                                                                    ;NO GO TO ERROR
 6222  025362  022705  000002          14$:  CMP     #2,R5                   ;IS R5 CORRECT
 6223  025366  001401                        BEQ     15$                     ;YES GO ON
 6224  025370  104001                        ERROR   .1                  ;CPU ERROR
 6225                                                                    ;NO GO TO ERROR
 6226                                  ;
 6227                                  ;
 6228                                  ;
 6229  025372  012701  025522          15$:  MOV     #TE1268,R1              ;SETUP POINTERS TO TABLES
 6230
 6231  025376  010137  110150          16$:  MOV     R1,@#EXPDAT             ;SAVE A COPY OF R1
```

```
6232  025402  011104                        MOV     (R1),R4         ;INIT R4
6233  025404  016103  000004                MOV     4(R1),R3                ;SAVE SOURCE
6234                                  ;
6235  025410  016105  000002                MOV     2(R1),R5                ;INIT R5
6236  025414  000277                        SCC                             ;SET ALL CC BITS
6237  025416  071461  000004                DIV     4(R1),R4                ; TEST INSTRUCTION
6238  025422  026137  000006  177776        CMP     6(R1),@#177776          ;IS PS CORRECT
6239  025430  001401                        BEQ     17$                     ;YES GO ON
6240  025432  104001                        ERROR   .1              ;CPU ERROR
6241                                                                ;NO GO TO ERROR
6242  025434  026105  000010          17$:  CMP     10(R1),R5               ;IS R5 CORRECT
6243  025440  001401                        BEQ     18$                     ;YES GO ON
6244  025442  104001                        ERROR   .1              ;CPU ERROR
6245                                                                ;NO GO TO ERROR
6246  025444  026104  000012          18$:  CMP     12(R1),R4               ;IS R4 CORRECT
6247  025450  001401                        BEQ     19$                     ;YES GO ON
6248  025452  104001                        ERROR   .1              ;CPU ERROR
6249                                                                ;NO GO TO ERROR
6250  025454  023701  110150          19$:  CMP     @#EXPDAT,R1             ;IS R1 CORRECT
6251  025460  001403                        BEQ     20$                     ;YES GO ON
6252  025462  104001                        ERROR   .1              ;CPU ERROR
6253                                                                ;NO GO TO ERROR
6254  025464  013701  110150                MOV     @#EXPDAT,R1             ;RESTORE CORRECT VALUE
6255  025470  026103  000004          20$:  CMP     4(R1),R3                ;IS SOURCE CORRECT
6256  025474  001403                        BEQ     21$                     ;YES GO ON
6257  025476  104001                        ERROR   .1              ;CPU ERROR
6258                                                                ;NO GO TO ERROR
6259  025500  010361  000004                MOV     R3,4(R1)                        ;TRY TO RESTORE CODE
6260  025504  062701  000014          21$:  ADD     #14,R1                  ;POINT TO NEXT LOCATION
6261  025510  021127  000333                CMP     (R1),#333               ;ARE WE DONE
6262  025514  001330                        BNE     16$                     ;NO GO TO 16$
6263
6264
6265  025516  000167  000316                JMP     FIN126
6266                                  ;
6267                                  ;
6268  025522  177777                TE1268: .WORD   177777  ;DIVIDEND
6269  025524  177777                        .WORD   177777  ;INIT R5
6270  025526  177777                        .WORD   177777  ;DIVISOR
6271  025530  000000                        .WORD   0       ;PSW
6272  025532  000000                        .WORD   0       ;R5 RESULT
6273  025534  000001                        .WORD   1       ;R4 RESULT
6274
6275  025536  000000                        .WORD   0       ;DIVIDEND
6276  025540  177777                        .WORD   177777  ;INIT R5
6277  025542  177777                        .WORD   177777  ;DIVISOR
6278  025544  000012                        .WORD   12      ;PSW
6279  025546  177777                        .WORD   177777  ;R5 RESULT
6280  025550  000000                        .WORD   0       ;R4 RESULT
6281
6282  025552  177777                        .WORD   177777  ;DIVIDEND
6283  025554  000000                        .WORD   0       ;INIT R5
6284  025556  177777                        .WORD   177777  ;DIVISOR
6285  025560  000002                        .WORD   2       ;PSW
6286  025562  000000                        .WORD   0       ;R5 RESULT
6287  025564  177777                        .WORD   177777  ;R4 RESULT
```

```
6288
6289  025566  000000                    .WORD   0       ;DIVIDEND
6290  025570  007642                    .WORD   7642    ;INIT R5
6291  025572  007643                    .WORD   7643    ;DIVISOR
6292  025574  000004                    .WORD   4       ;PSW
6293  025576  007642                    .WORD   7642    ;R5 RESULT
6294  025600  000000                    .WORD   0       ;R4 RESULT
6295
6296  025602  000000                    .WORD   0       ;DIVIDEND
6297  025604  000137                    .WORD   137     ;INIT R5
6298  025606  177543                    .WORD   177543  ;DIVISOR
6299  025610  000004                    .WORD   4       ;PSW
6300  025612  000137                    .WORD   137     ;R5 RESULT
6301  025614  000000                    .WORD   0       ;R4 RESULT
6302
6303  025616  000000                    .WORD   0       ;DIVIDEND
6304  025620  007643                    .WORD   7643    ;INIT R5
6305  025622  007643                    .WORD   7643    ;DIVISOR
6306  025624  000000                    .WORD   0       ;PSW
6307  025626  000000                    .WORD   0       ;R5 RESULT
6308  025630  000001                    .WORD   1       ;R4 RESULT
6309
6310  025632  100000                    .WORD   100000  ;DIVIDEND
6311  025634  004376                    .WORD   4376    ;INIT R5
6312  025636  010021                    .WORD   10021   ;DIVISOR
6313  025640  000012                    .WORD   12      ;PSW
6314  025642  004376                    .WORD   4376    ;R5 RESULT
6315  025644  100000                    .WORD   100000  ;R4 RESULT
6316
6317  025646  177700                    .WORD   177700  ;DIVIDEND
6318  025650  170033                    .WORD   170033  ;INIT R5
6319  025652  010021                    .WORD   10021   ;DIVISOR
6320  025654  000010                    .WORD   10      ;PSW
6321  025656  171307                    .WORD   171307  ;R5 RESULT
6322  025660  176024                    .WORD   176024  ;R4 RESULT
6323
6324  025662  177700                    .WORD   177700  ;DIVIDEND
6325  025664  170033                    .WORD   170033  ;INIT R5
6326  025666  167757                    .WORD   167757  ;DIVISOR
6327  025670  000000                    .WORD   0       ;PSW
6328  025672  171307                    .WORD   171307  ;R5 RESULT
6329  025674  001754                    .WORD   1754    ;R4 RESULT
6330
6331  025676  000000                    .WORD   0       ;DIVIDEND
6332  025700  177777                    .WORD   177777  ;INIT R5
6333  025702  000001                    .WORD   1       ;DIVISOR
6334  025704  000002                    .WORD   2       ;PSW
6335  025706  177777                    .WORD   177777  ;R5 RESULT
6336  025710  000000                    .WORD   0       ;R4 RESULT
6337
6338  025712  177777                    .WORD   177777  ;DIVIDEND
6339  025714  045716                    .WORD   45716   ;INIT R5
6340  025716  000001                    .WORD   1       ;DIVISOR
6341  025720  000012                    .WORD   12      ;PSW
6342  025722  045716                    .WORD   45716   ;R5 RESULT
6343  025724  177777                    .WORD   177777  ;R4 RESULT
```

```
6344
6345  025726  000000                          .WORD   0       ;DIVIDEND
6346  025730  000002                          .WORD   2       ;INIT R5
6347  025732  177770                          .WORD   177770  ;DIVISOR
6348  025734  000004                          .WORD   4       ;PSW
6349  025736  000002                          .WORD   2       ;R5 RESULT
6350  025740  000000                          .WORD   0       ;R4 RESULT
6351
6352  025742  177777                          .WORD   177777  ;DIVIDEND
6353  025744  177776                          .WORD   177776  ;INIT R5
6354  025746  000010                          .WORD   10      ;DIVISOR
6355  025750  000004                          .WORD   4       ;PSW
6356  025752  177776                          .WORD   177776  ;R5 RESULT
6357  025754  000000                          .WORD   0       ;R4 RESULT
6358
6359  025756  000001                          .WORD   1       ;DIVIDEND
6360  025760  177777                          .WORD   177777  ;INIT R5
6361  025762  000001                          .WORD   1       ;DIVISOR
6362  025764  000002                          .WORD   2       ;PSW
6363  025766  177777                          .WORD   177777  ;R5 RESULT
6364  025770  000001                          .WORD   1       ;R4 RESULT
6365
6366  025772  000001                          .WORD   1       ;DIVIDEND
6367  025774  000000                          .WORD   0       ;INIT R5
6368  025776  000002                          .WORD   2       ;DIVISOR
6369  026000  000002                          .WORD   2       ;PSW
6370  026002  000000                          .WORD   0       ;R5 RESULT
6371  026004  000001                          .WORD   1       ;R4 RESULT
6372
6373  026006  000001                          .WORD   1       ;DIVIDEND
6374  026010  000000                          .WORD   0       ;INIT R5
6375  026012  000003                          .WORD   3       ;DIVISOR
6376  026014  000000                          .WORD   0       ;PSW
6377  026016  000001                          .WORD   1       ;R5 RESULT
6378  026020  052525                          .WORD   52525   ;R4 RESULT
6379
6380  026022  000023                          .WORD   23      ;DIVIDEND
6381  026024  016054                          .WORD   16054   ;INIT R5
6382  026026  016537                          .WORD   16537   ;DIVISOR
6383  026030  000000                          .WORD   0       ;PSW
6384  026032  010222                          .WORD   10222   ;R5 RESULT
6385  026034  000246                          .WORD   246     ;R4 RESULT
6386
6387  026036  000333                          .WORD   333
6388  026040                          FIN126:
6389                                  ;
6390  026040                          TE127:
6391                                  ;       TEST ASH (ARITHMETIC SHIFT)
6392  026040  005037  177776                  CLR     @#177776                ;INIT PSW
6393  026044  012702  000001                  MOV     #1,R2                   ;SETUP OPERAND
6394  026050  000277                          SCC                             ;SET ALL CC BITS
6395  026052  072202                          ASH     R2,R2                   ; TEST INSTRUCTION
6396  026054  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
6397  026062  001401                          BEQ     1$                      ;YES GO ON
6398  026064  104001                          ERROR   +1              ;CPU ERROR
6399                                                                  ;NO GO TO ERROR
```

```
6400  026066  020227  000002          1$:     CMP     R2,#2                   ;IS R2 CORRECT
6401  026072  001401                          BEQ     2$                      ;YES GO ON
6402  026074  104001                          ERROR   .1              ;CPU ERROR
6403                                                                  ;NO GO TO ERROR
6404  026076  012702  100000          2$:     MOV     #100000,R2              ;SETUP R2
6405  026102  012703  000001                  MOV     #1,R3                   ;SETUP R3
6406  026106  000257                          CCC                             ;CLEAR ALL CC BITS
6407  026110  072203                          ASH     R3,R2                   ; TEST INSTRUCTION
6408  026112  022737  000007  177776          CMP     #7,@#177776             ;IS PS CORRECT
6409  026120  001401                          BEQ     3$                      ;YES GO ON
6410  026122  104001                          ERROR   .1              ;CPU ERROR
6411                                                                  ;NO GO TO ERROR
6412  026124  020327  000001          3$:     CMP     R3,#1                   ;IS R3 CORRECT
6413  026130  001401                          BEQ     4$                      ;YES GO ON
6414  026132  104001                          ERROR   .1              ;CPU ERROR
6415                                                                  ;NO GO TO ERROR
6416  026134  020227  000000          4$:     CMP     R2,#0                   ;IS R2 CORRECT
6417  026140  001401                          BEQ     5$                      ;YES GO ON
6418  026142  104001                          ERROR   .1              ;CPU ERROR
6419                                                                  ;NO GO TO ERROR
6420  026144  012701  026240          5$:     MOV     #TE127A,R1              ;SETUP POINTERS TO TABLES
6421
6422  026150  010103                  6$:     MOV     R1,R3                   ;
6423  026152  016102  000002                  MOV     2(R1),R2                ;SETUP R2
6424  026156  000277                          SCC                             ;SET ALL CC BITS
6425  026160  072211                          ASH     (R1),R2                 ; TEST INSTRUCTION
6426  026162  026137  000004  177776          CMP     4(R1),@#177776          ;IS PS CORRECT
6427  026170  001401                          BEQ     7$                      ;YES GO ON
6428  026172  104001                          ERROR   .1              ;CPU ERROR
6429                                                                  ;NO GO TO ERROR
6430  026174  026102  000006          7$:     CMP     6(R1),R2                ;IS R2 CORRECT
6431  026200  001401                          BEQ     8$                      ;YES GO ON
6432  026202  104001                          ERROR   .1              ;CPU ERROR
6433                                                                  ;NO GO TO ERROR
6434  026204  020301                  8$:     CMP     R3,R1                   ;IS R1 CORRECT
6435  026206  001402                          BEQ     9$                      ;YES GO ON
6436  026210  104001                          ERROR   .1              ;CPU ERROR
6437                                                                  ;NO GO TO ERROR
6438  026212  010301                          MOV     R3,R1                   ;RESTORE R1
6439  026214  021311                  9$:     CMP     (R3),(R1)               ;IS SOURCE CORRECT
6440  026216  001401                          BEQ     10$                     ;YES GO ON
6441  026220  104001                          ERROR   .1              ;CPU ERROR
6442                                                                  ;NO GO TO ERROR
6443                                                                  ;SOURCE LOOKS INCORRECT
6444  026222  062701  000010          10$:    ADD     #10,R1                  ;INCREMENT POINTER
6445  026226  020127  026500                  CMP     R1,#FIN127              ;ARE WE DONE
6446  026232  001346                          BNE     6$                      ;NO GO TO 6$
6447
6448
6449  026234  000167  000240                  JMP     FIN127
6450                                  ;
6451                                  ;
6452  026240  177761                  TE127A: .WORD   177761          ;SOURCE
6453  026242  077777                          .WORD   77777           ;DEST
6454  026244  000005                          .WORD   5
6455  026246  000000                          .WORD   0
```

```
6456
6457  026250  177700                  .WORD   177700          ;SOURCE
6458  026252  017777                  .WORD   17777           ;DEST
6459  026254  000000                  .WORD   0
6460  026256  017777                  .WORD   17777
6461
6462  026260  177700                  .WORD   177700          ;SOURCE
6463  026262  100000                  .WORD   100000          ;DEST
6464  026264  000010                  .WORD   10
6465  026266  100000                  .WORD   100000
6466
6467  026270  177777                  .WORD   177777          ;SOURCE
6468  026272  100000                  .WORD   100000          ;DEST
6469  026274  000010                  .WORD   10
6470  026276  140000                  .WORD   140000
6471
6472  026300  177737                  .WORD   177737          ;SOURCE
6473  026302  177777                  .WORD   177777          ;DEST
6474  026304  000011                  .WORD   11
6475  026306  177777                  .WORD   177777
6476
6477  026310  177706                  .WORD   177706          ;SOURCE
6478  026312  102000                  .WORD   102000          ;DEST
6479  026314  000007                  .WORD   7
6480  026316  000000                  .WORD   0
6481
6482  026320  177710                  .WORD   177710          ;SOURCE
6483  026322  017777                  .WORD   17777           ;DEST
6484  026324  000013                  .WORD   13
6485  026326  177400                  .WORD   177400
6486
6487  026330  177713                  .WORD   177713          ;SOURCE
6488  026332  000012                  .WORD   12              ;DEST
6489  026334  000000                  .WORD   0
6490  026336  050000                  .WORD   50000
6491
6492  026340  177707                  .WORD   177707          ;SOURCE
6493  026342  170001                  .WORD   170001          ;DEST
6494  026344  000002                  .WORD   2
6495  026346  000200                  .WORD   200
6496
6497  026350  177717                  .WORD   177717          ;SOURCE
6498  026352  000001                  .WORD   1               ;DEST
6499  026354  000012                  .WORD   12
6500  026356  100000                  .WORD   100000
6501
6502  026360  177740                  .WORD   177740          ;SOURCE
6503  026362  017777                  .WORD   17777           ;DEST
6504  026364  000004                  .WORD   4
6505  026366  000000                  .WORD   0
6506
6507  026370  177771                  .WORD   177771          ;SOURCE
6508  026372  150000                  .WORD   150000          ;DEST
6509  026374  000010                  .WORD   10
6510  026376  177640                  .WORD   177640
6511
```

```
6512  026400  177742                          .WORD   177742          ;SOURCE
6513  026402  100000                          .WORD   100000          ;DEST
6514  026404  000011                          .WORD   11
6515  026406  177777                          .WORD   177777
6516
6517  026410  177764                          .WORD   177764          ;SOURCE
6518  026412  100000                          .WORD   100000          ;DEST
6519  026414  000010                          .WORD   10
6520  026416  177770                          .WORD   177770
6521
6522  026420  177750                          .WORD   177750          ;SOURCE
6523  026422  052525                          .WORD   52525           ;DEST
6524  026424  000004                          .WORD   4
6525  026426  000000                          .WORD   0
6526
6527  026430  177760                          .WORD   177760          ;SOURCE
6528  026432  100000                          .WORD   100000          ;DEST
6529  026434  000011                          .WORD   11
6530  026436  177777                          .WORD   177777
6531
6532  026440  177770                          .WORD   177770          ;SOURCE
6533  026442  100000                          .WORD   100000          ;DEST
6534  026444  000010                          .WORD   10
6535  026446  177600                          .WORD   177600
6536
6537  026450  177712                          .WORD   177712          ;SOURCE
6538  026452  004367                          .WORD   4367            ;DEST
6539  026454  000013                          .WORD   13
6540  026456  156000                          .WORD   156000
6541
6542  026460  177764                          .WORD   177764          ;SOURCE
6543  026462  017777                          .WORD   17777           ;DEST
6544  026464  000001                          .WORD   1
6545  026466  000001                          .WORD   1
6546
6547  026470  177701                          .WORD   177701          ;SOURCE
6548  026472  110000                          .WORD   110000          ;DEST
6549  026474  000003                          .WORD   3
6550  026476  020000                          .WORD   20000
6551
6552  026500  000240              FIN127: NOP
6553                              ;
6554  026502                      TE130:
6555                              ;       TEST ASHC (ARITHMETIC SHIFT COMBINED)
6556  026502  005037  177776              CLR     @#177776                ;INIT PSW
6557  026506  012701  000023              MOV     #23,R1                  ;SETUP R1
6558  026512  012705  052525              MOV     #52525,R5               ;SETUP R5
6559  026516  005004                      CLR     R4                      ;SETUP R4
6560  026520  000277                      SCC                             ;SET ALL CC BITS
6561  026522  073401                      ASHC    R1,R4                   ; TEST INSTRUCTION
6562  026524  023727  177776  000012      CMP     @#177776,#12            ;IS PS CORRECT
6563  026532  001401                      BEQ     1$                      ;YES GO ON
6564  026534  104001                      ERROR   ^1              ;CPU ERROR
6565                                                              ;NO GO TO ERROR
6566  026536  020127  000023      1$:     CMP     R1,#23                  ;IS R1 CORRECT
6567  026542  001401                      BEQ     2$                      ;YES GO ON
```

```
6568  026544  104001                          ERROR   ;1              ;CPU ERROR
6569                                                                  ;NO GO TO ERROR
6570  026546  020427  125250          2$:     CMP     R4,#125250      ;IS R4 CORRECT
6571  026552  001401                          BEQ     3$              ;YES GO ON
6572  026554  104001                          ERROR   ;1              ;CPU ERROR
6573                                                                  ;NO GO TO ERROR
6574  026556  020527  000000          3$:     CMP     R5,#0           ;IS R5 CORRECT
6575  026562  001401                          BEQ     4$              ;YES GO ON
6576  026564  104001                          ERROR   ;1              ;CPU ERROR
6577                                                                  ;NO GO TO ERROR
6578  026566  012703  052525          4$:     MOV     #52525,R3       ;SETUP R3
6579  026572  005002                          CLR     R2              ;SETUP R2
6580  026574  012704  164731                  MOV     #164731,R4      ;SETUP R4
6581  026600  000277                          SCC                     ;SET ALL CC BITS
6582  026602  073327  000023                  ASHC    #23,R3          ; TEST INSTRUCTION
6583  026606  023727  177776  000012          CMP     @#177776,#12    ;IS PS CORRECT
6584  026614  001401                          BEQ     5$              ;YES GO ON
6585  026616  104001                          ERROR   ;1              ;CPU ERROR
6586                                                                  ;NO GO TO ERROR
6587  026620  020227  000000          5$:     CMP     R2,#0           ;IS R2 CORRECT
6588  026624  001401                          BEQ     6$              ;YES GO ON
6589  026626  104001                          ERROR   ;1              ;CPU ERROR
6590                                                                  ;NO GO TO ERROR
6591  026630  020327  000000          6$:     CMP     R3,#0           ;IS R3 CORRECT
6592  026634  001401                          BEQ     7$              ;YES GO ON
6593  026636  104001                          ERROR   ;1              ;CPU ERROR
6594                                                                  ;NO GO TO ERROR
6595  026640  020427  164731          7$:     CMP     R4,#164731      ;IS R4 CORRECT
6596  026644  001401                          BEQ     8$              ;YES GO ON
6597  026646  104001                          ERROR   ;1              ;CPU ERROR
6598                                                                  ;NO GO TO ERROR
6599                                                                  ;
6600                                                                  ;
6601  026650  012701  026760          8$:     MOV     #TE130A,R1      ;SETUP POINTERS TO TABLES
6602
6603  026654  010104                  9$:     MOV     R1,R4           ;SAVE A COPY OF R1
6604  026656  016102  000002                  MOV     2(R1),R2        ;SETUP R2
6605  026662  016103  000004                  MOV     4(R1),R3        ;SETUP R3
6606  026666  000277                          SCC                     ;SET ALL CC BITS
6607  026670  073211                          ASHC    (R1),R2         ; TEST INSTRUCTION
6608  026672  023761  177776  000006          CMP     @#177776,6(R1)  ;IS PS CORRECT
6609  026700  001401                          BEQ     10$             ;YES GO ON
6610  026702  104001                          ERROR   ;1              ;CPU ERROR
6611                                                                  ;NO GO TO ERROR
6612  026704  026102  000010          10$:    CMP     10(R1),R2       ;IS R2 CORRECT
6613  026710  001401                          BEQ     11$             ;YES GO ON
6614  026712  104001                          ERROR   ;1              ;CPU ERROR
6615                                                                  ;NO GO TO ERROR
6616  026714  026103  000012          11$:    CMP     12(R1),R3       ;IS R3 CORRECT
6617  026720  001401                          BEQ     12$             ;YES GO ON
6618  026722  104001                          ERROR   ;1              ;CPU ERROR
6619                                                                  ;NO GO TO ERROR
6620  026724  020401                  12$:    CMP     R4,R1           ;IS R1 CORRECT
6621  026726  001402                          BEQ     13$             ;YES GO ON
6622  026730  104001                          ERROR   ;1              ;CPU ERROR
6623                                                                  ;NO GO TO ERROR
```

```
6624  026732  010401                    MOV     R4,R1
6625  026734  021114                13$:  CMP     (R1),(R4)           ;IS SOURCE CORRECT
6626  026736  001401                    BEQ     14$                 ;YES GO ON
6627  026740  104001                    ERROR   .1                  ;CPU ERROR
6628                                                                ;NO GO TO ERROR
6629                                                                ;POSSIBLE SOURCE CODE CORRUPTION
6630  026742  062701  000014        14$:  ADD     #14,R1              ;GO TO NEXT TEST
6631  026746  020127  027370            CMP     R1,#FIN130          ;ARE WE DONE
6632  026752  001340                    BNE     9$                  ;NO GO TO 9$
6633
6634
6635  026754  000167  000410            JMP     FIN130
6636                                ;
6637                                ;
6638  026760  177700            TE130A: .WORD   177700              ;SOURCE
6639  026762  100125                    .WORD   100125              ;DESTINATION WORD 1
6640  026764  177777                    .WORD   177777              ;DESTINATION WORD 2
6641  026766  000010                    .WORD   10                  ;TEST PSW
6642  026770  100125                    .WORD   100125              ;RESULT WORD 1
6643  026772  177777                    .WORD   177777              ;RESULT WORD 2
6644
6645  026774  177777                    .WORD   177777              ;SOURCE
6646  026776  000001                    .WORD   1                   ;DESTINATION WORD 1
6647  027000  000000                    .WORD   0                   ;DESTINATION WORD 2
6648  027002  000000                    .WORD   0                   ;TEST PSW
6649  027004  000000                    .WORD   0                   ;RESULT WORD 1
6650  027006  100000                    .WORD   100000              ;RESULT WORD 2
6651
6652  027010  177701                    .WORD   177701              ;SOURCE
6653  027012  047777                    .WORD   47777               ;DESTINATION WORD 1
6654  027014  100000                    .WORD   100000              ;DESTINATION WORD 2
6655  027016  000012                    .WORD   12                  ;TEST PSW
6656  027020  117777                    .WORD   117777              ;RESULT WORD 1
6657  027022  000000                    .WORD   0                   ;RESULT WORD 2
6658
6659  027024  177706                    .WORD   177706              ;SOURCE
6660  027026  004256                    .WORD   4256                ;DESTINATION WORD 1
6661  027030  177700                    .WORD   177700              ;DESTINATION WORD 2
6662  027032  000002                    .WORD   2                   ;TEST PSW
6663  027034  025677                    .WORD   25677               ;RESULT WORD 1
6664  027036  170000                    .WORD   170000              ;RESULT WORD 2
6665
6666  027040  177711                    .WORD   177711              ;SOURCE
6667  027042  065700                    .WORD   65700               ;DESTINATION WORD 1
6668  027044  000012                    .WORD   12                  ;DESTINATION WORD 2
6669  027046  000013                    .WORD   13                  ;TEST PSW
6670  027050  100000                    .WORD   100000              ;RESULT WORD 1
6671  027052  012000                    .WORD   12000               ;RESULT WORD 2
6672
6673  027054  177737                    .WORD   177737              ;SOURCE
6674  027056  000000                    .WORD   0                   ;DESTINATION WORD 1
6675  027060  000001                    .WORD   1                   ;DESTINATION WORD 2
6676  027062  000004                    .WORD   4                   ;TEST PSW
6677  027064  000000                    .WORD   0                   ;RESULT WORD 1
6678  027066  000000                    .WORD   0                   ;RESULT WORD 2
6679
```

```
 6680  027070  177736                    .WORD   177736          ;SOURCE
 6681  027072  000000                    .WORD   0               ;DESTINATION WORD 1
 6682  027074  000001                    .WORD   1               ;DESTINATION WORD 2
 6683  027076  000000                    .WORD   0               ;TEST PSW
 6684  027100  040000                    .WORD   40000           ;RESULT WORD 1
 6685  027102  000000                    .WORD   0               ;RESULT WORD 2
 6686
 6687  027104  177740                    .WORD   177740          ;SOURCE
 6688  027106  100000                    .WORD   100000          ;DESTINATION WORD 1
 6689  027110  000000                    .WORD   0               ;DESTINATION WORD 2
 6690  027112  000011                    .WORD   11              ;TEST PSW
 6691  027114  177777                    .WORD   177777          ;RESULT WORD 1
 6692  027116  177777                    .WORD   177777          ;RESULT WORD 2
 6693
 6694  027120  177725                    .WORD   177725          ;SOURCE
 6695  027122  177777                    .WORD   177777          ;DESTINATION WORD 1
 6696  027124  174000                    .WORD   174000          ;DESTINATION WORD 2
 6697  027126  000007                    .WORD   7               ;TEST PSW
 6698  027130  000000                    .WORD   0               ;RESULT WORD 1
 6699  027132  000000                    .WORD   0               ;RESULT WORD 2
 6700
 6701  027134  177724                    .WORD   177724          ;SOURCE
 6702  027136  177777                    .WORD   177777          ;DESTINATION WORD 1
 6703  027140  174000                    .WORD   174000          ;DESTINATION WORD 2
 6704  027142  000011                    .WORD   11              ;TEST PSW
 6705  027144  100000                    .WORD   100000          ;RESULT WORD 1
 6706  027146  000000                    .WORD   0               ;RESULT WORD 2
 6707
 6708  027150  177733                    .WORD   177733          ;SOURCE
 6709  027152  177777                    .WORD   177777          ;DESTINATION WORD 1
 6710  027154  157023                    .WORD   157023          ;DESTINATION WORD 2
 6711  027156  000012                    .WORD   12              ;TEST PSW
 6712  027160  114000                    .WORD   114000          ;RESULT WORD 1
 6713  027162  000000                    .WORD   0               ;RESULT WORD 2
 6714
 6715  027164  177727                    .WORD   177727          ;SOURCE
 6716  027166  000000                    .WORD   0               ;DESTINATION WORD 1
 6717  027170  177777                    .WORD   177777          ;DESTINATION WORD 2
 6718  027172  000013                    .WORD   13              ;TEST PSW
 6719  027174  177600                    .WORD   177600          ;RESULT WORD 1
 6720  027176  000000                    .WORD   0               ;RESULT WORD 2
 6721
 6722  027200  177717                    .WORD   177717          ;SOURCE
 6723  027202  177777                    .WORD   177777          ;DESTINATION WORD 1
 6724  027204  000001                    .WORD   1               ;DESTINATION WORD 2
 6725  027206  000011                    .WORD   11              ;TEST PSW
 6726  027210  100000                    .WORD   100000          ;RESULT WORD 1
 6727  027212  100000                    .WORD   100000          ;RESULT WORD 2
 6728
 6729  027214  177741                    .WORD   177741          ;SOURCE
 6730  027216  100000                    .WORD   100000          ;DESTINATION WORD 1
 6731  027220  000000                    .WORD   0               ;DESTINATION WORD 2
 6732  027222  000010                    .WORD   10              ;TEST PSW
 6733  027224  177777                    .WORD   177777          ;RESULT WORD 1
 6734  027226  177777                    .WORD   177777          ;RESULT WORD 2
 6735
```

```
 6736  027230  177742                  .WORD   177742          ;SOURCE
 6737  027232  037777                  .WORD   37777           ;DESTINATION WORD 1
 6738  027234  177777                  .WORD   177777          ;DESTINATION WORD 2
 6739  027236  000005                  .WORD   5               ;TEST PSW
 6740  027240  000000                  .WORD   0               ;RESULT WORD 1
 6741  027242  000000                  .WORD   0               ;RESULT WORD 2
 6742
 6743  027244  177742                  .WORD   177742          ;SOURCE
 6744  027246  077777                  .WORD   77777           ;DESTINATION WORD 1
 6745  027250  177777                  .WORD   177777          ;DESTINATION WORD 2
 6746  027252  000001                  .WORD   1               ;TEST PSW
 6747  027254  000000                  .WORD   0               ;RESULT WORD 1
 6748  027256  000001                  .WORD   1               ;RESULT WORD 2
 6749
 6750  027260  177711                  .WORD   177711          ;SOURCE
 6751  027262  065600                  .WORD   65600           ;DESTINATION WORD 1
 6752  027264  000012                  .WORD   12              ;DESTINATION WORD 2
 6753  027266  000003                  .WORD   3               ;TEST PSW
 6754  027270  000000                  .WORD   0               ;RESULT WORD 1
 6755  027272  012000                  .WORD   12000           ;RESULT WORD 2
 6756
 6757  027274  177740                  .WORD   177740          ;SOURCE
 6758  027276  077777                  .WORD   77777           ;DESTINATION WORD 1
 6759  027300  177777                  .WORD   177777          ;DESTINATION WORD 2
 6760  027302  000004                  .WORD   4               ;TEST PSW
 6761  027304  000000                  .WORD   0               ;RESULT WORD 1
 6762  027306  000000                  .WORD   0               ;RESULT WORD 2
 6763
 6764  027310  177737                  .WORD   177737          ;SOURCE
 6765  027312  177777                  .WORD   177777          ;DESTINATION WORD 1
 6766  027314  177774                  .WORD   177774          ;DESTINATION WORD 2
 6767  027316  000011                  .WORD   11              ;TEST PSW
 6768  027320  177777                  .WORD   177777          ;RESULT WORD 1
 6769  027322  177777                  .WORD   177777          ;RESULT WORD 2
 6770
 6771  027324  177747                  .WORD   177747          ;SOURCE
 6772  027326  100000                  .WORD   100000          ;DESTINATION WORD 1
 6773  027330  174000                  .WORD   174000          ;DESTINATION WORD 2
 6774  027332  000010                  .WORD   10              ;TEST PSW
 6775  027334  177777                  .WORD   177777          ;RESULT WORD 1
 6776  027336  177700                  .WORD   177700          ;RESULT WORD 2
 6777
 6778  027340  177753                  .WORD   177753          ;SOURCE
 6779  027342  006324                  .WORD   6324            ;DESTINATION WORD 1
 6780  027344  071002                  .WORD   71002           ;DESTINATION WORD 2
 6781  027346  000001                  .WORD   1               ;TEST PSW
 6782  027350  000000                  .WORD   0               ;RESULT WORD 1
 6783  027352  000146                  .WORD   146             ;RESULT WORD 2
 6784
 6785  027354  177765                  .WORD   177765          ;SOURCE
 6786  027356  102351                  .WORD   102351          ;DESTINATION WORD 1
 6787  027360  177231                  .WORD   177231          ;DESTINATION WORD 2
 6788  027362  000011                  .WORD   11              ;TEST PSW
 6789  027364  177760                  .WORD   177760          ;RESULT WORD 1
 6790  027366  116477                  .WORD   116477          ;RESULT WORD 2
 6791  027370              FIN130:
```

```
                        ************ DOUBLE OPERAND TESTS ******************                    SEQ 0125

 6792  027370                       MSPAU:
 6793
 6794                               ;       TEST THAT AUTO DEC/INC OPERATIONS USING SP ARE ON WORD BOUNDRYS
 6795  027370  005006                       CLR     R6                      ;CLEAR SP
 6796  027372  112667  153462               MOVB    (R6)+,COUNT             ;TRY AUTOINC ON R6
 6797  027376  022706  000002               CMP     #2,R6                   ;VERIFY AUTO INC BY 2
 6798  027402  001401                       BEQ     SPAU1                   ;BRANCH IF GOOD
 6799                                                                       ;BAD AUTO-INC
 6800  027404  104001                       ERROR   +1              ;CPU ERROR
 6801  027406  005006               SPAU1:  CLR     R6                      ;CLEAR R6
 6802  027410  112667  153444               MOVB    (R6)+,COUNT
 6803  027414  112667  153440               MOVB    (R6)+,COUNT             ;DOUBLE BYTE AUTO-INC
 6804  027420  022706  000004               CMP     #4,R6           ;VERIFY RESULT
 6805  027424  001401                       BEQ     SPAU2           ;BRANCH IF GOOD
 6806  027426  104001                       ERROR   +1              ;CPU ERROR
 6807                                                   ;BAD DOUBLE AUTO-INC
 6808  027430  012706  001000       SPAU2:  MOV     #STBOT,R6               ;LOAD R6
 6809  027434  114667  153420               MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6810  027440  022706  000776               CMP     #776,R6         ;VERIFY RESULT
 6811  027444  001401                       BEQ     SPAU3           ;BRANCH IF GOOD
 6812  027446  104001                       ERROR   +1              ;CPU ERROR
 6813
 6814  027450  012706  001000       SPAU3:  MOV     #STBOT,R6               ;LOAD R6
 6815  027454  114667  153400               MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6816  027460  114667  153374               MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6817  027464  022706  000774               CMP     #774,R6         ;VERIFY RESULT
 6818  027470  001401                       BEQ     SPAU4           ;BRANCH IF GOOD
 6819  027472  104001                       ERROR   +1              ;CPU ERROR
 6820
 6821  027474  005006               SPAU4:  CLR     R6              ; TEST AUTO-INC ON SOP
 6822  027476  105726                       TSTB    (R6)+           ; TEST AUTO-INC
 6823  027500  020627  000002               CMP     R6,#2
 6824  027504  001401                       BEQ     SPAU5           ;BRANCH IF GOOD
 6825  027506  104001                       ERROR   +1              ;CPU ERROR
 6826
 6827  027510  012706  001000       SPAU5:  MOV     #STBOT,R6       ;LOAD R6
 6828  027514  105746                       TSTB    -(R6)           ; TEST AUTO-DEC
 6829  027516  022706  000776               CMP     #776,R6         ;VERIFY RESULT
 6830  027522  001401                       BEQ     SPAU6           ;BRANCH IF GOOD
 6831  027524  104001                       ERROR   +1              ;CPU ERROR
 6832
 6833  027526  012706  001000       SPAU6:  MOV     #STBOT,R6
 6834
 6835                               ;
 6836  027532                       MTRY:
 6837
 6838                               ;       VERIFY YELLOW ZONE TRAP ON AUTO DEC OF R6
 6839  027532  005067  150230               CLR     CPEREG          ;INIT CPU ERROR REGISTER
 6840  027536  012706  000150               MOV     #150,R6         ;LOAD R6 WITH A VALUE THAT WILL
 6841                                                               ;CAUSE A YELLOW STACK TRAP(IE. <400)
 6842  027542  016767  150236  153212       MOV     4,SLOC00        ;SAVE VECTOR
 6843  027550  012767  027606  150226       MOV     #MTRYA,4        ;SETUP THE STACK OVERFLOW TRAP POINTER
 6844  027556  016701  150364               MOV     146,R1          ;SAVE VECTOR
 6845  027562  016702  150356               MOV     144,R2          ;SAVE VECTOR
 6846  027566  016703  150350               MOV     142,R3          ;SAVE VECTOR
 6847  027572  005067  150350               CLR     146             ;JUST AS A PRECAUTION
```

```
 6848  027576  005046                          CLR     -(R6)           ;CAUSE A STACK OVERFLOW TRAP
 6849  027600  012706  001000                  MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
 6850  027604  104001                          ERROR   +1              ;CPU ERROR
 6851                                                          ;OVERFLOW TRAP FAILED
 6852  027606                          MTRYA:
 6853  027606  022767  000010  150152          CMP     #BIT03,CPEREG   ;WAS CPU ERROR REG SET PROPERLY?
 6854  027614  001003                          BNE     1$              ;GO TO ERROR IF NOT
 6855  027616  020627  000142                  CMP     R6,#142         ;VERIFY CORRECT DECRIMENT OF R6
 6856  027622  001401                          BEQ     MTRYB           ;BRANCH IF GOOD
 6857  027624  104001                  1$:     ERROR   +1              ;CPU ERROR
 6858                                                          ;R6 IMPROPERLY DECRIMENTED
 6859                                                          ;OR CPU ERROR REGISTER NOT CORRECT
 6860  027626                          MTRYB:
 6861  027626  005067  150134                  CLR     CPEREG          ;CLEAR THE CPU ERROR REGISTER
 6862  027632  016767  153124  150144          MOV     SLOC00,4        ;RESTORE VECTOR
 6863  027640  010167  150302                  MOV     R1,146          ;RESTORE VECTORS
 6864  027644  010267  150274                  MOV     R2,144          ;       "
 6865  027650  010367  150266                  MOV     R3,142          ;       "
 6866  027654  012706  001000                  MOV     #STBOT,R6
 6867
 6868
 6869                                  ;
 6870  027660                          MTRYM:
 6871
 6872                                  ;       TEST STACK OVERFLOW TRAPS IN VARIOUS MODES
 6873  027660  005067  150102                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6874  027664  012706  000400                  MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6875  027670  016767  150110  153064          MOV     4,SLOC00                ;SAVE VECTOR
 6876  027676  012767  027720  150100          MOV     #TRYMA,4
 6877  027704  005067  150466                  CLR     376                     ;JUST AS A PRECAUTION
 6878  027710  005046                          CLR     -(R6)                   ;CAUSE OVERFLOW TRAP
 6879  027712  012706  001000                  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6880  027716  104001                          ERROR   +1              ;CPU ERROR
 6881                                                                  ;NO OVERFLOW TRAP
 6882  027720                          TRYMA:
 6883  027720  005067  150042                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6884  027724  012705  001000                  MOV     #1000,R5                ;SETUP R5 DATA
 6885  027730  012706  000400                  MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6886  027734  012767  027752  150042          MOV     #TRYMB,4
 6887  027742  064645                          ADD     -(R6),-(R5)             ;CAUSE OVERFLOW TRAP
 6888  027744  012706  001000                  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6889  027750  104001                          ERROR   +1              ;CPU ERROR
 6890                                                                  ;NO OVERFLOW TRAP
 6891  027752                          TRYMB:
 6892  027752  005067  150010                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6893  027756  012706  000150                  MOV     #150,R6                 ;SETUP OVERFLOW R6 DATA
 6894  027762  012767  030000  150014          MOV     #TRYMC,4
 6895  027770  044546                          BIC     -(R5),-(R6)             ;CAUSE OVERFLOW TRAP
 6896  027772  012706  001000                  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6897  027776  104001                          ERROR   +1              ;CPU ERROR
 6898                                                                  ;NO OVERFLOW TRAP
 6899  030000  005067  147762          TRYMC:  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6900  030004  016767  152752  147772          MOV     SLOC00,4                ;RESTORE VECTOR
 6901  030012  012706  001000                  MOV     #STBOT,R6
 6902
 6903
```

```
                                        ;
030016                                  MILLO:
                                        ;       TEST STACK OVERFLOW ON ILLEGAL INST TRAP
030016  005067  147744                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030022  012706  000400                  MOV     #400,R6                 ;SETUP FOR OVERFLOW TRAP
030026  016767  147756  152726          MOV     10,SLOC00               ;SAVE VECTOR
030034  012767  030062  147746          MOV     #MILLOA,10              ;SETUP ILLEGAL TRAP VECTOR
030042  016767  147736  152714          MOV     4,SLOC01                ;SAVE VECTOR
030050  012767  030070  147726          MOV     #MILLOB,4               ;SETUP OVERFLOW TRAP VECTOR
030056  000077                          77                              ;UNUSED INSTRUCTION TRAP
030060  000240                          NOP
030062  012706  001000          MILLOA: MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030066  104001                          ERROR   +1              ;CPU ERROR
                                                                ;UNUSED INSTRUCTION TRAP
030070                          MILLOB:
030070  016767  152670  147706          MOV     SLOC01,4                ;RESTORE VECTOR
030076  016767  152660  147704          MOV     SLOC00,10               ;RESTORE VECTOR
030104  005067  147656                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030110  012706  001000                  MOV     #STBOT,R6               ;RESTORE R6
                                        ;
030114                                  MIOTO:
                                        ;       TEST STACK OVERFLOW ON IOT TRAP
030114  005067  147646                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030120  012706  000400                  MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
030124  016767  147670  152630          MOV     20,SLOC00                       ;SAVE OLD IOT VECTOR
030132  012767  030160  147660          MOV     #IOTOA,20                       ;SETUP ERROR ACTION ON IOT
030140  016767  147640  152616          MOV     4,SLOC01                        ;SAVE VECTOR
030146  012767  030166  147630          MOV     #IOTOB,4                        ;SETUP CORRECT TRAP VECTOR FOR
                                                                                ;OVERFLOW
030154  000004                          IOT                                     ; TEST INSTRUCTION
030156  000240                          NOP
030160  012706  001000          IOTOA:  MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
030164  104001                          ERROR   +1              ;CPU ERROR
                                                                ;FAILURE OF STACK OVERFLOW
030166                          IOTOB:
030166  005067  147574                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030172  012706  001000                  MOV     #STBOT,R6
030176  016767  152562  147600          MOV     SLOC01,4                        ;RESTORE VECTOR
030204  016767  152552  147606          MOV     SLOC00,20                       ;RESTORE TRAP VECTOR
                                        ;
                                        ;
030212                                  MEMTO:
                                        ;       TEST STACK OVERFLOW ON EMT TRAP
030212  005067  147550                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030216  012706  000400                  MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
030222  016767  147602  152532          MOV     30,SLOC00                       ;SAVE OLD EMT VECTOR
030230  012767  030256  147572          MOV     #EMTOA,30                       ;SETUP ERROR ACTION ON EMT
030236  016767  147542  152520          MOV     4,SLOC01                        ;SAVE VECTOR
030244  012767  030264  147532          MOV     #EMTOB,4                        ;SETUP CORRECT TRAP VECTOR FOR
```

```
                                                                        ;OVERFLOW
030252  104000                          EMT                             ; TEST INSTRUCTION
030254  000240                          NOP
030256  012706  001000          EMTOA:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030262  104001                          ERROR   +1              ;CPU ERROR
                                                                ;FAILURE OF STACK OVERFLOW
030264                          EMTOB:
030264  016767  152472  147536        MOV     SLOC00,30               ;RESTORE TRAP VECTOR
030272  016767  152466  147504        MOV     SLOC01,4                ;RESTORE VECTOR
030300  005067  147462                CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
030304  012706  001000                MOV     #STBOT,R6

030310                          MTRP0:

                        ;       TEST STACK OVERFLOW ON TRAP
030310  005067  147452                CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
030314  012706  000400                MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
030320  016767  147510  152434        MOV     34,SLOC00               ;SAVE OLD TRP VECTOR
030326  012767  030354  147500        MOV     #TRPOA,34               ;SETUP ERROR ACTION ON TRP
030334  016767  147444  152422        MOV     4,SLOC01                ;SAVE VECTOR
030342  012767  030362  147434        MOV     #TRPOB,4                ;SETUP CORRECT TRAP VECTOR FOR
                                                                ;OVERFLOW
030350  104400                        TRAP                            ; TEST INSTRUCTION
030352  000240                        NOP
030354  012706  001000          TRPOA:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030360  104001                        ERROR   +1              ;CPU ERROR
                                                                ;FAILURE OF STACK OVERFLOW
030362                          TRPOB:
030362  016767  152374  147444        MOV     SLOC00,34               ;RESTORE TRAP VECTOR
030370  016767  152370  147406        MOV     SLOC01,4                ;RESTORE VECTOR
030376  005067  147364                CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
030402  012706  001000                MOV     #STBOT,R6

                        ;
                        ;
030406                          MBPTO:

                        ;       TEST STACK OVERFLOW ON BPT
030406  005067  147354                CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
030412  012706  000400                MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
030416  016767  147372  152336        MOV     14,SLOC00               ;SAVE OLD BPT VECTOR
030424  012767  030452  147362        MOV     #BPTOA,14               ;SETUP ERROR ACTION ON BPT
030432  016767  147346  152324        MOV     4,SLOC01                ;SAVE VECTOR
030440  012767  030460  147336        MOV     #BPTOB,4                ;SETUP CORRECT TRAP VECTOR FOR
                                                                ;OVERFLOW
030446  000003                        BPT                             ; TEST INSTRUCTION
030450  000240                        NOP
030452  012706  001000          BPTOA:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030456  104001                        ERROR   +1              ;CPU ERROR
                                                                ;FAILURE OF STACK OVERFLOW
030460                          BPTOB:
030460  005067  147302                CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
030464  016767  152272  147322        MOV     SLOC00,14               ;RESTORE TRAP VECTOR
030472  016767  152266  147304        MOV     SLOC01,4                ;RESTORE VECTOR
030500  012706  001000                MOV     #STBOT,R6
```

```
7016                                    ;
7017                                    ;
7018  030504                            MILAO:
7019
7020                                    ;       TEST STACK OVERFLOW AND ILLEGAL JMP INSTRUCTION
7021  030504  005067  147256                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7022  030510  012706  000400                      MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
7023  030514  016767  147270  152240              MOV     10,SLOC00                       ;SAVE OLD ILLEGAL INST. VECTOR
7024  030522  012767  030552  147260              MOV     #ILAOA,10                       ;SETUP ERROR ACTION ILLEGAL OPCODE
7025  030530  016767  147250  152226              MOV     4,SLOC01                        ;SAVE VECTOR
7026  030536  012767  030560  147240              MOV     #ILB08,4                        ;SETUP CORRECT TRAP VECTOR FOR
7027                                                                                    ;OVERFLOW
7028  030544  005001                            CLR     R1
7029  030546  000101                            JMP     R1                              ; TEST INSTRUCTION
7030  030550  000240                            NOP
7031  030552  012706  001000            ILAOA:  MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
7032  030556  104001                            ERROR   +1              ;CPU ERROR
7033                                                                                    ;FAILURE OF STACK OVERFLOW
7034  030560                            ILB08:
7035  030560  016767  152200  147216              MOV     SLOC01,4                        ;RESTORE VECTOR
7036  030566  016767  152170  147214              MOV     SLOC00,10                       ;RESTORE TRAP VECTOR
7037  030574  005067  147166                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7038  030600  012706  001000                      MOV     #STBOT,R6
7039
7040                                    ;
7041                                    ;
7042  030604                            MILLBO:
7043
7044                                    ;       TEST STACK OVERFLOW ON ILLEGAL JSR INST
7045  030604  012706  000400                      MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
7046  030610  0.6767  147174  152144              MOV     10,SLOC00                       ;SAVE OLD VECTOR
7047  030616    2767  030646  147164              MOV     #ILLBOA,10                      ;SETUP ERROR ACTION ON ILL. OPCODE
7048  030624  016767  147154  152132              MOV     4,SLOC01                        ;SAVE VECTOR
7049  030632  012767  030654  147144              MOV     #ILLB08,4                       ;SETUP CORRECT TRAP VECTOR FOR
7050                                                                                    ;OVERFLOW
7051  030640  005001                            CLR     R1
7052  030642  004501                            JSR     R5,R1                           ; TEST INSTRUCTION
7053  030644  000240                            NOP
7054  030646  012706  001000            ILLBOA: MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
7055  030652  104001                            ERROR   +1              ;CPU ERROR
7056                                                                                    ;FAILURE OF STACK OVERFLOW
7057  030654                            ILLB08:
7058  030654  016767  152104  147122              MOV     SLOC01,4                        ;RESTORE VECTOR
7059  030662  016767  152074  147120              MOV     SLOC00,10                       ;RESTORE TRAP VECTOR
7060  030670  012706  001000                      MOV     #STBOT,R6
7061
7062                                    ;
7063                                    ;
7064  030674                            MSTO:
7065
7066                                    ;       TEST FOR FALSE STACK OVERFLOW
7067  030674  016767  147104  152060              MOV     4,SLOC00                ;SAVE VECTOR
7068  030702  012767  030750  147074              MOV     #MSTOE,4                ;ANTICIPATE OVERFLOW ERROR
7069  030710  012706  001002                      MOV     #1002,R6                ;SETUP LEGAL R6
7070  030714  005746                            TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7071  030716  012706  002002                      MOV     #2002,R6                ;SETUP LEGAL R6
```

```
7072  030722  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7073  030724  012706  004002                  MOV     #4002,R6                ;SETUP LEGAL R6
7074  030730  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7075  030732  012706  010002                  MOV     #10002,R6               ;SETUP LEGAL R6
7076  030736  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7077  030740  012706  100402                  MOV     #100402,R6              ;SETUP  LEGAL R6
7078  030744  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7079  030746  000403                          BR      MSTOEE                  ;EXIT MODULE
7080  030750  012706  001000          MSTOE:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
7081  030754  104001                          ERROR   +1              ;CPU ERROR
7082                                                                  ;STACK OVERFLOW ERROR
7083  030756  016767  152000  147020  MSTOEE: MOV     SLOC00,4                ;RESTORE VECTOR
7084  030764  012706  001000                  MOV     #STBOT,R6
7085
7086                                  ;
7087                                  ;
7088  030770                          MTT:
7089
7090                                  ;       TEST T-BIT TRAPS
7091  030770  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7092  030774  016767  147014  151760          MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7093  031002  012746  000020                  MOV     #20,-(R6)               ;PUSH T-BIT
7094  031006  012746  031024                  MOV     #MTTA,-(R6)             ;SETUP ERROR TRAP VECTOR
7095  031012  012767  031026  146774          MOV     #MTTB,14                ;SETUP NEW T-BIT VECTOR
7096  031020  000002                          RTI                             ;CAUSE A T BIT SET IN PSW
7097  031022  104001                          ERROR   +1              ;CPU ERROR
7098                                                                  ;SHOULD NEVER BE EXECUTED
7099  031024  104001          MTTA:   ERROR   +1              ;CPU ERROR
7100                                                                  ;DIDNT TAKE CORRECT TRAP
7101  031026  022706  000774          MTTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
7102  031032  001401                          BEQ     MTTD                    ;BRANCH IF GOOD
7103  031034  104001                          ERROR   +1              ;CPU ERROR
7104                                                                  ;BAD SP
7105  031036  021627  031024          MTTD:   CMP     (R6),#MTTA              ;VERIFY PC SAVED ON STACK
7106  031042  001401                          BEQ     MTTE                    ;BRANCH IF GOOD
7107  031044  104001                          ERROR   +1              ;CPU ERROR
7108                                                                  ;INCORRECT PC ON STACK
7109  031046                          MTTE:
7110  031046  016767  151710  146740          MOV     SLOC00,14               ;RESTORE VECTOR 14
7111
7112  031054  012706  001000                  MOV     #STBOT,R6
7113
7114                                  ;
7115  031060                          MTTS.
7116
7117                                  ;       TEST T-BIT TRAPS WITH RTT
7118  031060  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7119  031064  016767  146724  151670          MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7120  031072  012746  000020                  MOV     #20,-(R6)               ;PUSH T-BIT
7121  031076  012746  031114                  MOV     #MTTSA,-(R6)            ;SETUP ERROR TRAP VECTOR
7122  031102  012767  031120  146704          MOV     #MTTSB,14               ;SETUP NEW T-BIT VECTOR
7123  031110  000006                          RTT                             ;CAUSE A T BIT SET IN PSW
7124  031112  104001                          ERROR   +1              ;CPU ERROR
7125                                                                  ;SHOULD NEVER BE EXECUTED
7126  031114  000240          MTTSA:  NOP                             ;RTT WILL EXECUTE THIS INSTRUCTION
7127                                                                  ;WITH A T-BIT TRAP
```

```
7128  031116  104001                  MTTSQ:  ERROR   ;1              ;CPU ERROR
7129                                                                  ;DIDNT TAKE CORRECT TRAP
7130  031120  022706  000774          MTTSB:  CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
7131  031124  001401                          BEQ     MTTSD                   ;BRANCH IF GOOD
7132  031126  104001                          ERROR   ;1              ;CPU ERROR
7133                                                                  ;BAD SP
7134  031130  021627  031116          MTTSD:  CMP     (R6),#MTTSQ             ;VERIFY PC SAVED ON STACK
7135  031134  001401                          BEQ     MTTSE                   ;BRANCH IF GOOD
7136  031136  104001                          ERROR   ;1              ;CPU ERROR
7137                                                                  ;INCORRECT PC ON STACK
7138  031140                          MTTSE:
7139  031140  016767  151616  146646        MOV     SLOC00,14               ;RESTORE VECTOR 14
7140  031146  012706  001000                MOV     #STBOT,R6
7141
7142  031152                          MTTR:
7143
7144                                  ;       TEST OLD STATUS ON T-BIT TRAP
7145  031152  012706  001000                MOV     #STBOT,R6               ;SETUP STACK
7146  031156  016767  146632  151576        MOV     14,SLOC00               ;SAVE OLD  VECTOR
7147  031164  012746  000020                MOV     #20,-(R6)               ;PUSH T-BIT
7148  031170  012746  031216                MOV     #MTTRA,-(R6)            ;SETUP ERROR TRAP VECTOR
7149  031174  012767  031220  146612        MOV     #MTTRB,14               ;SETUP NEW T-BIT VECTOR
7150  031202  012767  000357  146566        MOV     #357,PS                 ;SET PRIORITY AND COND C
7151  031210  000277                        SCC
7152  031212  000002                        RTI
7153  031214  104001                        ERROR   ;1              ;CPU ERROR
7154                                                                  ;SHOULD NEVER EXECUTE
7155  031216  104001                  MTTRA:  ERROR   ;1              ;CPU ERROR
7156                                                                  ;DIDNT TAKE CORRECT TRAP
7157  031220  026727  147552  000020  MTTRB:  CMP     STBOT-2,#20             ;VERIFY PSW ON STACK
7158  031226  001401                          BEQ     MTTRC                   ;BRANCH IF CORRECT STATUS
7159  031230  104001                          ERROR   ;1              ;CPU ERROR
7160                                                                  ;BAD STATUS ON STACK
7161  031232  012706  001000          MTTRC:  MOV     #STBOT,R6               ;SETUP STACK
7162  031236  012746  000377                MOV     #377,-(R6)              ;PUSH T-BIT
7163  031242  012746  031270                MOV     #MTTRD,-(R6)            ;SETUP ERROR TRAP VECTOR
7164  031246  012767  031272  146540        MOV     #MTTRE,14               ;SETUP NEW T-BIT VECTOR
7165  031254  012767  000000  146514        MOV     #0,PS                   ;CLEAR PRIORITY
7166  031262  000257                        CCC                             ;CLEAR CONDITION CODES
7167  031264  000002                        RTI
7168  031266  104001                        ERROR   ;1              ;CPU ERROR
7169                                                                  ;SHOULD NEVER EXECUTE
7170  031270  104001                  MTTRD:  ERROR   ;1              ;CPU ERROR
7171                                                                  ;DIDNT TAKE CORRECT TRAP
7172  031272  026727  147500  000377  MTTRE:  CMP     STBOT-2,#377            ;VERIFY OLD PSW ON STACK
7173  031300  001401                          BEQ     MTTRF                   ;BRANCH IF GOOD
7174  031302  104001                          ERROR   ;1              ;CPU ERROR
7175                                                                  ;OLD PSW INCORRECT
7176  031304                          MTTR:
7177  031304  016767  151452  146502        MOV     SLOC00,14               ;RESTORE VECTOR
7178  031312  012706  001000                MOV     #STBOT,R6
7179
7180                                  ;
7181  031316                          MRT:
7182
7183                                  ;       TEST RESERVED INST TRAP
```

```
7184  031316  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7185  031322  016767  146462  151432          MOV     10,SLOC00               ;SAVE OLD  VECTOR
7186  031330  012767  031342  146452          MOV     #MRTB,10                ;SETUP NEW RESERVED VECTOR
7187  031336  000077                          77
7188  031340  104001                  MRTA:   ERROR   .1              ;CPU ERROR
7189                                                                  ;DIDNT TAKE CORRECT TRAP
7190  031342  022706  000774          MRTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7191  031346  001401                          BEQ     MRTE                    ;BRANCH IF GOOD
7192  031350  104001                          ERROR   .1              ;CPU ERROR
7193                                                                  ;BAD PC ON STACK
7194  031352  021627  031340          MRTE:   CMP     (R6),#MRTA              ;VERFY PROPER PC ON STACK
7195  031356  001401                          BEQ     MRTF                    ;BRANCH IF GOOD
7196  031360  104001                          ERROR   .1              ;CPU ERROR
7197                                                                  ;INCORRECT PC ON STACK
7198  031362                          MRTF:
7199  031362  016767  151374  146420          MOV     SLOC00,10               ;RESTORE TRAP VECTOR
7200  031370  012706  001000                  MOV     #STBOT,R6
7201
7202                                  ;
7203                                  ;
7204  031374                          MRTO:
7205
7206                                  ;       TEST OLD STATUS ON RESERVED INST TRAP
7207  031374  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7208  031400  016767  146404  151354          MOV     10,SLOC00               ;SAVE OLD  VECTOR
7209  031406  012767  031426  146374          MOV     #MRTOB,10               ;SETUP NEW VECTOR
7210  031414  005067  146356                  CLR     PS                      ;CLEAR PRIORITY AND COND C
7211  031420  000257                          CCC
7212  031422  000077                          77
7213  031424  104001                  MRTOA:  ERROR   .1              ;CPU ERROR
7214                                                                  ;DIDNT TAKE CORRECT TRAP
7215  031426  026727  147344  000000  MRTOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7216  031434  001401                          BEQ     MRTOC                   ;BRANCH IF CORRECT STATUS
7217  031436  104001                          ERROR   .1              ;CPU ERROR
7218                                                                  ;BAD STATUS ON STACK
7219  031440  012706  001000          MRTOC:  MOV     #STBOT,R6               ;SETUP STACK
7220  031444  012767  031466  146336          MOV     #MRTOE,10               ;SET UP TRAP VECTOR
7221  031452  012767  000357  146316          MOV     #357,PS                 ;SET PRIORITY
7222  031460  000277                          SCC                             ;SET CONDITION CODES
7223  031462  000077                          77                              ;RESERVED INSTRUCTION
7224
7225  031464  104001                  MRTOD:  ERROR   .1              ;CPU ERROR
7226                                                                  ;DIDNT TAKE CORRECT TRAP
7227  031466  026727  147304  000357  MRTOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7228  031474  001401                          BEQ     MRTOF                   ;BRANCH IF GOOD
7229  031476  104001                          ERROR   .1              ;CPU ERROR
7230                                                                  ;OLD PSW INCORRECT
7231  031500                          MRTOF:
7232  031500  016767  151256  146302          MOV     SLOC00,10               ;RESOTRE TRAP VECTOR
7233  031506  012706  001000                  MOV     #STBOT,R6
7234
7235  031512                          MTP:
7236
7237                                  ;       TEST TRAP INST
7238  031512  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7239  031516  016767  146312  151236          MOV     34,SLOC00               ;SAVE OLD  VECTOR
```

```
7240  031524  012767  031544  146302          MOV     #MTPB,34                ;SETUP NEW TRAP VECTOR
7241  031532  005067  146240                  CLR     PS                      ;CLEAR PRIORITY ABND COND C
7242  031536  000257                          CCC
7243  031540  104400                          TRAP
7244  031542  104001                  MTPR:   ERROR   +1              ;CPU ERROR
7245                                                                  ;DIDNT TAKE CORRECT TRAP
7246  031544  022706  000774          MTPB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7247  031550  001401                          BEQ     MTPQ                    ;BRANCH IF GOOD
7248  031552  104001                          ERROR   +1              ;CPU ERROR
7249                                                                  ;BAD PC ON STACK
7250  031554  021627  031542          MTPQ:   CMP     (R6),#MTPR              ;VERFY PROPER PC ON STACK
7251  031560  001401                          BEQ     MTPF                    ;BRANCH IF GOOD
7252  031562  104001                          ERROR   +1              ;CPU ERROR
7253                                                                  ;INCORRECT PC ON STACK
7254  031564                          MTPF:
7255  031564  016767  151172  146242          MOV     SLOC00,34               ;RESTORE VECTOR
7256  031572  012706  001 00                  MOV     #STBOT,R6
7257
7258                                  ;
7259                                  ;
7260  031576                          MTPO:
7261
7262                                  ;       TEST OLD STATUS SAVED ON TRAP
7263  031576  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7264  031602  016767  146226  151152          MOV     34,SLOC00               ;SAVE OLD  VECTOR
7265  031610  012767  031630  146216          MOV     #MTPOB,34               ;SETUP NEW TRAP VECTOR
7266  031616  005067  146154                  CLR     PS                      ;CLEAR PRIORITY AND COND C
7267  031622  000257                          CCC
7268  031624  104400                          TRAP
7269  031626  104001                  MTPOA:  ERROR   +1              ;CPU ERROR
7270                                                                  ;DIDNT TAKE CORRECT TRAP
7271  031630  026727  147142  000000  MTPOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7272  031636  001401                          BEQ     MTPOC                   ;BRANCH IF CORRECT STATUS
7273  031640  104001                          ERROR   +1              ;CPU ERROR
7274                                                                  ;BAD STATUS ON STACK
7275  031642  012706  001000          MTPOC:  MOV     #STBOT,R6               ;SETUP STACK
7276  031646  012767  031670  146160          MOV     #MTPOE,34               ;SET UP TRAP VECTOR
7277  031654  012767  000357  146114          MOV     #357,PS                 ;SET PRIORITY
7278  031662  000277                          SCC                             ;SET CONDITION CODES
7279  031664  104400                          TRAP                            ;ISSUE TRAP
7280  031666  104001                  MTPOD:  ERROR   +1              ;CPU ERROR
7281                                                                  ;DIDNT TAKE CORRECT TRAP
7282  031670  026727  147102  000357  MTPOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7283  031676  001401                          BEQ     MTPOF                   ;BRANCH IF GOOD
7284  031700  104001                          ERROR   +1              ;CPU ERROR
7285                                                                  ;OLD PSW INCORRECT
7286  031702                          MTPOF:
7287  031702  016767  151054  146124          MOV     SLOC00,34               ;RESTORE TRAP VECTOR
7288  031710  012706  001000                  MOV     #STBOT,R6
7289
7290                                  ;
7291                                  ;
7292  031714                          MTPA:
7293
7294                                  ;       TEST ALL TRAP OPCODES - SELF MODIFYING
7295  031714  005003                          CLR     R3                      ;SETUP REGISTER TO INDICATE OPCODE
```

```
 7296  031716  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7297  031722  016767  146106  151032          MOV     34,SLOC00               ;SAVE OLD  VECTOR
 7298  031730  016767  146050  151026          MOV     4,SLOC01                ;SAVE IN CASE OF HALT
 7299  031736  012767  031766  146040          MOV     #MTPAH,4                ;SETUP HALT TRAP
 7300  031744  012767  031770  146062          MOV     #MTPAA,34               ;SETUP NEW TRAP VECTOR
 7301  031752  000167  000012                  JMP     MTPAA                   ;GO INTO LOOPING CODE
 7302                                  ;
 7303  031756  000000                  MTPAL:  HALT                            ;SET TO A ZERO
 7304  031760  104001                          ERROR   .1              ;CPU ERROR
 7305                                          ;TRAP INSTRUCTION FAILED TO TRAP
 7306                                                  ;EXAMINE OPCCODE AT LOCATION MTPAL:
 7307  031762  000167  000002                  JMP     MTPAA                   ;ATTEMPT TO GO ON
 7308                                  ;
 7309  031766  104001                  MTPAH:  ERROR   .1              ;CPU ERROR
 7310                                          ;ERROR, EITHER CANT MODIFY LOCATION MTPAL
 7311                                                  ;OR TRAP INSTRUCTION FAILED
 7312  031770                          MTPAA:
 7313  031770  005203                          INC     R3                      ;GET NEXT OPCODE
 7314
 7315  031772  012706  001000                  MOV     #STBOT,R6               ;RESTORE STACK
 7316  031776  020327  000400                  CMP     R3,#400                 ;SEE IF LAST OPCODE
 7317  032002  001406                          BEQ     MTPAE                   ;BRANCH IF DONE
 7318  032004  012767  104400  177744          MOV     #104400,MTPAL           ;TRAP OPCODE INTO LOCATION
 7319  032012  060367  177740                  ADD     R3,MTPAL                ;FORM TEST OPCODE
 7320  032016  000757                          BR      MTPAL                   ;EXECUTE TEST
 7321  032020                          MTPAE:
 7322
 7323  032020  016767  150736  146006          MOV     SLOC00,34
 7324  032026  016767  150732  145750          MOV     SLOC01,4                ;RESTORE VECTORS
 7325
 7326  032034  012706  001000                  MOV     #STBOT,R6
 7327
 7328                                  ;
 7329                                  ;
 7330  032040                          MIOT:
 7331
 7332                                  ;       TEST IOT TRAP
 7333  032040  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7334  032044  016767  145750  150710          MOV     20,SLOC00               ;SAVE OLD  VECTOR
 7335  032052  012767  032064  145740          MOV     #MIOTB,20               ;SETUP NEW IOT VECTOR
 7336  032060  000004                          IOT
 7337  032062  104001                  MIOTA:  ERROR   .1              ;CPU ERROR
 7338                                          ;DIDNT TAKE CORRECT TRAP
 7339  032064  022706  000774          MIOTB:  CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
 7340  032070  001401                          BEQ     MIOTD                   ;BRANCH IF GOOD
 7341  032072  104001                          ERROR   .1              ;CPU ERROR
 7342                                          ;BAD PC ON STACK
 7343  032074  021627  032062          MIOTD:  CMP     (R6),#MIOTA             ;VERFY PROPER PC ON STACK
 7344  032100  001401                          BEQ     MIOTF                   ;BRANCH IF GOOD
 7345  032102  104001                          ERROR   .1              ;CPU ERROR
 7346                                          ;INCORRECT PC ON STACK
 7347  032104  016767  150652  145706  MIOTF:  MOV     SLOC00,20               ;RESTORE VECTOR
 7348  032112  012706  001000                  MOV     #STBOT,R6
 7349
 7350  032116                          MITO:
 7351
```

```
7352                                    ;       TEST OLD STATUS ON IOT TRAP
7353  032116  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7354  032122  016767  145672  150632            MOV     20,SLOC00               ;SAVE OLD  VECTOR
7355  032130  012767  032150  145662            MOV     #MITOB,20               ;SETUP NEW IOT VECTOR
7356  032136  005067  145634                    CLR     PS                      ;CLEAR PRIORITY AND COND C
7357  032142  000257                            CCC
7358  032144  000004                            IOT
7359  032146  104001                MITOA:      ERROR   .1              ;CPU ERROR
7360                                                                    ;DIDNT TAKE CORRECT TRAP
7361  032150  026727  146622  000000  MITOB:    CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7362  032156  001401                            BEQ     MITOC                   ;BRANCH IF CORRECT STATUS
7363  032160  104001                            ERROR   .1              ;CPU ERROR
7364                                                                    ;BAD STATUS ON STACK
7365  032162  012706  001000        MITOC:      MOV     #STBOT,R6               ;SETUP STACK
7366  032166  012767  000340  145624            MOV     #MITOE,20               ;SET UP TRAP VECTOR
7367  032174  012767  000357  145574            MOV     #357,PS                 ;SET PRIORITY
7368  032202  000277                            SCC                             ;SET CONDITION CODES
7369  032204  000004                            IOT
7370  032206  104001                MITOD:      ERROR   .1              ;CPU ERROR
7371                                                                    ;DIDNT TAKE CORRECT TRAP
7372  032210  026727  146562  000357  MITOE:    CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7373  032216  001401                            BEQ     MITOF                   ;BRANCH IF GOOD
7374  032220  104001                            ERROR   .1              ;CPU ERROR
7375                                                                    ;OLD PSW INCORRECT
7376  032222                        MITOF:
7377  032222  016767  150534  145570            MOV     SLOC00,20               ;RESTORE VECTOR
7378  032230  012706  001000                    MOV     #STBOT,R6
7379
7380                                    ;
7381  032234                        MET:
7382
7383                                    ;       TEST EMULATOR TRAP INSTRUCTION  (EMT)
7384  032234  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7385  032240  016767  145564  150514            MOV     30,SLOC00               ;SAVE OLD  VECTOR
7386  032246  012767  032302  145554            MOV     #METB,30                ;SETUP NEW EMT VECTOR
7387  032254  016767  145554  150502            MOV     34,SLOC01               ;SAVE TRAP VECTOR
7388  032262  012767  133342  145544            MOV     #1ERROR,34              ;SET UP TO HANDLE EMT ERROR
7389  032270  104000                            EMT
7390  032272  104400                META:       TRAP                            ;TRAP ON ERROR
7391  032274  001057                            .WORD   559.
7392  032276  000001                            .WORD   1                       ;CPUERR
7393  032300  000001                            .WORD   1                       ;ERRTN
7394                                                                  ;DIDNT TAKE CORRECT TRAP
7395  032302  022706  000774        METB:       CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7396  032306  001401                            BEQ     METD                    ;BRANCH IF GOOD
7397  032310  104001                            ERROR   .1              ;CPU ERROR
7398                                                                  ;BAD PC ON STACK
7399  032312  021627  032272        METD:       CMP     (R6),#META              ;VERFY PROPER PC ON STACK
7400  032316  001401                            BEQ     METF                    ;BRANCH IF GOOD
7401  032320  104001                            ERROR   .1              ;CPU ERROR
7402                                                                  ;INCORRECT PC ON STACK
7403  032322  016767  150436  145504  METF:     MOV     SLOC01,34               ;RESTORE VECTOR
7404  032330  016767  150426  145472            MOV     SLOC00,30               ;RESTORE VECTOR
7405  032336  012706  001000                    MOV     #STBOT,R6
7406
7407                                    ;
```

```
 7408  032342                                  METO:
 7409
 7410                                          ;       TEST OLD STATUS ON EMT 104401
 7411  032342  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7412  032346  016767  145456  150406          MOV     30,SLOC00               ;SAVE OLD  VECTOR
 7413  032354  012767  032416  145446          MOV     #METOB,30               ;SETUP NEW EMT VECTOR
 7414  032362  016767  145446  150374          MOV     34,SLOC01               ;SAVE TRAP VECTOR
 7415  032370  012767  133342  145436          MOV     #340,34                 ;SET UP TRAP VECTOR
 7416  032376  005067  145374                  CLR     PS                      ;CLEAR PRIORITY AND COND C
 7417  032402  000257                          CCC
 7418  032404  104000                          EMT
 7419  032406  104400                  METOA:  TRAP
 7420  032410  001062                          .WORD   562.
 7421  032412  000001                          .WORD   1                               ;CPUERR
 7422  032414  000001                          .WORD   1                               ;ERRTN
 7423                                                                  ;DIDNT TAKE CORRECT TRAP
 7424  032416  026727  146354  000000  METOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
 7425  032424  001401                          BEQ     METOC                   ;BRANCH IF CORRECT STATUS
 7426  032426  104001                          ERROR   .1                      ;CPU ERROR
 7427                                                                  ;BAD STATUS ON STACK
 7428  032430  012706  001000          METOC:  MOV     #STBOT,R6               ;SETUP STACK
 7429  032434  012767  032464  145366          MOV     #METOE,30               ;SET UP TRAP VECTOR
 7430  032442  012767  000357  145326          MOV     #357,PS                 ;SET PRIORITY
 7431  032450  000277                          SCC                             ;SET CONDITION CODES
 7432  032452  104000                          EMT
 7433  032454  104400                  METOD:  TRAP
 7434  032456  001064                          .WORD   564.
 7435  032460  000001                          .WORD   1                               ;CPUERR
 7436  032462  000001                          .WORD   1                               ;ERRTN
 7437                                                                  ;DIDNT TAKE CORRECT TRAP
 7438  032464  026727  146306  000357  METOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
 7439  032472  001401                          BEQ     METOF                   ;BRANCH IF GOOD
 7440  032474  104001                          ERROR   .1                      ;CPU ERROR
 7441                                                                  ;OLD PSW INCORRECT
 7442  032476                          METOF:
 7443  032476  016767  150262  145330          MOV     SLOC01,34               ;RESTORE VECTOR
 7444  032504  016767  150252  145316          MOV     SLOC00,30               ;RESTORE VECTOR
 7445  032512  012706  001000                  MOV     #STBOT,R6
 7446
 7447                                          ;
 7450                                          ;
 7449  032516                          MBT:
 7450
 7451                                          ;       TEST BPT TRAP
 7452  032516  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7453  032522  016767  145266  150232          MOV     14,SLOC00               ;SAVE OLD  VECTOR
 7454  032530  012767  032542  145256          MOV     #MBTB,14                ;SETUP NEW BPT VECTOR
 7455  032536  000003                          BPT
 7456  032540  104001                  MBTA:   ERROR   .1                      ;CPU ERROR
 7457                                                                  ;DIDNT TAKE CORRECT TRAP
 7458  032542  022706  000774          MBTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
 7459  032546  001401                          BEQ     MBTD                    ;BRANCH IF GOOD
 7460  032550  104001                          ERROR   .1                      ;CPU ERROR
 7461                                                                  ;BAD PC ON STACK
 7462  032552  021627  032540          MBTD:   CMP     (R6),#MBTA              ;VERFY PROPER PC ON STACK
 7463  032556  001401                          BEQ     MBTF                    ;BRANCH IF GOOD
```

```
7464  032560  104001                          ERROR   ;1              ;CPU ERROR
7465                                                                  ;INCORRECT PC ON STACK
7466  032562  016767  150174  145224  MBTF:   MOV     SLOC00,14               ;RESTORE VECTOR
7467  032570  012706  001000                  MOV     #STBOT,R6
7468
7469                                  ;
7470                                  ;
7471  032574                          MBTO:
7472
7473                                  ;       TEST OLD STATUS ON BPT TRAP
7474  032574  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7475  032600  016767  145210  150154          MOV     14,SLOC00               ;SAVE OLD  VECTOR
7476  032606  012767  032626  145200          MOV     #MBT08,14               ;SETUP NEW BPT VECTOR
7477  032614  005067  145156                  CLR     PS                      ;CLEAR PRIORITY AND COND C
7478  032620  000257                          CCC
7479  032622  000003                          BPT
7480  032624  104001                  MBTOA:  ERROR   ;1              ;CPU ERROR
7481                                                                  ;DIDNT TAKE CORRECT TRAP
7482  032626  026727  146144  000000  MBT08:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7483  032634  001401                          BEQ     MBTOC           ;BRANCH IF CORRECT STATUS
7484  032636  104001                          ERROR   ;1              ;CPU ERROR
7485                                                                  ;BAD STATUS ON STACK
7486  032640  012706  001000          MBTOC:  MOV     #STBOT,R6               ;SETUP STACK
7487  032644  012767  032666  145142          MOV     #MBTOE,14               ;SET UP TRAP VECTOR
7488  032652  012767  000357  145116          MOV     #357,PS                 ;SET PRIORITY
7489  032660  000277                          SCC                             ;SET CONDITION CODES
7490  032662  000003                          BPT
7491  032664  104001                  MBTOD:  ERROR   ;1              ;CPU ERROR
7492                                                                  ;DIDNT TAKE CORRECT TRAP
7493  032666  026727  146104  000357  MBTOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7494  032674  001401                          BEQ     MBTOF                   ;BRANCH IF GOOD
7495  032676  104001                          ERROR   ;1              ;CPU ERROR
7496                                                                  ;OLD PSW INCORRECT
7497  032700                          MBTOF:
7498  032700  016767  150056  145106          MOV     SLOC00,14               ;RESTORE VECTOR
7499  032706  012706  001000                  MOV     #STBOT,R6
7500
7501                                  ;
7502                                  ;
7503  032712                          MIL:
7504
7505                                  ;       TEST ILLEGAL JUMP INSTRUCTION TRAP
7506  032712  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7507  032716  016767  145066  150036          MOV     10,SLOC00               ;SAVE OLD  VECTOR
7508  032724  012767  032740  145056          MOV     #MILB,10                ;SETUP NEW ILLEGAL VECTOR
7509  032732  005001                          CLR     R1
7510  032734  000101                          JMP     R1                      ;**TEST INSTRUCTIO
7511  032736  104001                  MILA:   ERROR   ;1              ;CPU ERROR
7512                                                                  ;DIDNT TAKE CORRECT TRAP
7513  032740  022706  000774          MILB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7514  032744  001401                          BEQ     MILD                    ;BRANCH IF GOOD
7515  032746  104001                          ERROR   ;1              ;CPU ERROR
7516                                                                  ;BAD PC ON STACK
7517  032750  021627  032736          MILD:   CMP     (R6),#MILA              ;VERFY PROPER PC ON STACK
7518  032754  001401                          BEQ     MILF                    ;BRANCH IF GOOD
7519  032756  104001                          ERROR   ;1              ;CPU ERROR
```

```
7520                                                            ;INCORRECT PC ON STACK
7521  032760  016767  147776  :45022  MILF:   MOV     SLOC00,10               ;RESTORE VECTOR
7522  032766  012706  001000          MOV     #STBOT,R6
7523
7524  032772                         MILO:
7525
7526                                 ;       TEST OLD STATUS ON ILLEGAL JUMP TRAP
7527  032772  012706  001000          MOV     #STBOT,R6               ;SETUP STACK
7528  032776  016767  145006  147756  MOV     10,SLOC00               ;SAVE OLD  VECTOR
7529  033004  012767  033026  144776  MOV     #MILOB,10               ;SETUP NEW ILLEGAL VECTOR
7530  033012  005067  144760          CLR     PS                      ;CLEAR PRIORITY AND COND C
7531  033016  000257                  CCC
7532  033020  005001                  CLR     R1
7533  033022  000101                  JMP     R1
7534  03302*  104001          MILOA:  ERROR   ·1              ;CPU ERROR
7535                                                          ;DIDNT TAKE CORRECT TRAP
7536  033026  026727  145744  000004  MILOB:  CMP     STBOT-2,#4              ;VERIFY PSW ON STACK
7537  033034  001401                  BEQ     MILOC                   ;BRANCH IF CORRECT STATUS
7538  033036  104001                  ERROR   ·1              ;CPU ERROR
7539                                                          ;BAD STATUS ON STACK
7540  033040  012706  001000          MILOC:  MOV     #STBOT,R6               ;SETUP STACK
7541  033044  012767  033066  144736  MOV     #MILOE,10               ;SET UP TRAP VECTOR
7542  033052  012767  000357  144716  MOV     #357,PS                 ;SET PRIORITY
7543  0*3060  000277                  SCC                             ;SET CONDITION CODES
7544  033062  000101                  JMP     R1
7545  033064  104001          MILOD:  ERROR   ·1              ;CPU ERROR
7546                                                          ;DIDNT TAKE CORRECT TRAP
7547  033066  026727  145704  000357  MILOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7548  033074  001401                  BEQ     MILOF                   ;BRANCH IF GOOD
7549  033076  104001                  ERROR   ·1              ;CPU ERROR
7550                                                          ;OLD PSW INCORRECT
7551  033100                         MILOF:
7552  033100  016767  147656  144702          MOV     SLOC00,10               ;RESTORE VECTOR
7553  033106  012706  001000          MOV     #STBOT,R6
7554
7555                                 ;
7556                                 ;
7557  033112                         MIALL:
7558
7559                                 ;       TEST ILLEGAL JSR INSTRUCTION TRAP
7560  033112  012706  001000          MOV     #STBOT,R6               ;SETUP STACK
7561  033116  016767  144666  147636  MOV     10,SLOC00               ;SAVE OLD  VECTOR
7562  033124  012767  033140  144656  MOV     #MIALLB,10              ;SETUP NEW ILLEGAL VECTOR
7563  033132  005003                  CLR     R3
7564  033134  004303                  JSR     R3,R3
7565  033136  104001          MIALLA: ERROR   ·1              ;CPU ERROR
7566                                                          ;DIDNT TAKE CORRECT TRAP
7567  033140  022706  000774  MIALLB: CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7568  033144  001401                  BEQ     MIALLD                  ;BRANCH IF GOOD
7569  033146  104001                  ERROR   ·1              ;CPU ERROR
7570                                                          ;BAD PC ON STACK
7571  033150  021627  033136  MIALLD: CMP     (R6),#MIALLA            ;VERFY PROPER PC ON STACK
7572  033154  001401                  BEQ     MIALLF                  ;BRANCH IF GOOD
7573  033156  104001                  ERROR   ·1              ;CPU ERROR
7574                                                          ;INCORRECT PC ON STACK
7575  033160  016767  147576  144622  MIALLF: MOV     SLOC00,10               ;RESTORE VECTOR
```

```
 7576  033166  012706  001000                  MOV     #STBOT,R6
 7577
 7578                                  ;
 7579                                  ;
 7580  033172                          MJSI:
 7581
 7582                                  ;       TEST OLD STATUS ON ILLEGAL JSR TRAP
 7583  033172  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7584  033176  016767  144606  147556          MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7585  033204  012767  033226  144576          MOV     #MJSIB,10               ;SETUP NEW VECTOR
 7586  033212  005067  144560                  CLR     PS                      ;CLEAR PRIORITY AND COND C
 7587  033216  000257                          CCC
 7588  033220  005003                          CLR     R3
 7589  033222  004303                          JSR     R3,R3
 7590  033224  104001                  MJSIA:  ERROR   .1                      ;CPU ERROR
 7591                                                                          ;DIDNT TAKE CORRECT TRAP
 7592  033226  026727  145544  000004  MJSIB:  CMP     STBOT-2,#4                      ;VERIFY PSW ON STACK
 7593  033234  001401                          BEQ     MJSIC                   ;BRANCH IF CORRECT STATUS
 7594  033236  104001                          ERROR   .1                      ;CPU ERROR
 7595                                                                          ;BAD STATUS ON STACK
 7596  033240  012706  001000          MJSIC:  MOV     #STBOT,R6               ;SETUP STACK
 7597  033244  012767  033266  144536          MOV     #MJSIE,10               ;SET UP TRAP VECTOR
 7598  033252  012767  000357  144516          MOV     #357,PS                 ;SET PRIORITY
 7599  033260  000277                          SCC                             ;SET CONDITION CODES
 7600  033262  004303                          JSR     R3,R3
 7601  033264  104001                  MJSID:  ERROR   .1                      ;CPU ERROR
 7602                                                                          ;DIDNT TAKE CORRECT TRAP
 7603  033266  026727  145504  000357  MJSIE:  CMP     STBOT-2,#357                    ;VERIFY OLD PSW ON STACK
 7604  033274  001401                          BEQ     MJSIF                   ;BRANCH IF GOOD
 7605  033276  104001                          ERROR   .1                      ;CPU ERROR
 7606                                                                          ;OLD PSW INCORRECT
 7607  033300                          MJSIF:
 7608  033300  016767  147456  144502          MOV     SLOC00,10               ;RESTORE VECTOR
 7609  033306  012706  001000                  MOV     #STBOT,R6
 7610
 7611                                  ;
 7612                                  ;
 7613  033312                          IOXXX:
 7614                                  ;       I/O TIME OUT TEST
 7615  033312  005067  144450                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7616  033316  016767  144462  147436          MOV     4,SLOC00                ;SAVE VECTOR
 7617  033324  012767  033346  144452          MOV     #2$,4                   ;SET UP VECTOR TO HANDLE NXM
 7618  033332  012767  030000  144436          MOV     #30000,PS               ;INIT THE PSW TO A KNOWN STATE
 7619  033340  005737  177700                  TST     @#177700                ;TRY TO ACCESS HARDWARE ADDRESS
 7620                                                                          ;FOR GENERAL PURPOSE REG 0. THIS
 7621                                                                          ;IS NOT IMPLEMENTED ON KDJ11
 7622                                                                          ;SHOULD CAUSE TIME OUT.
 7623  033344  104001                  1$:     ERROR   .1                      ;CPU ERROR
 7624  033346  022767  000020  144412  2$:     CMP     #BIT04,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
 7625  033354  001401                          BEQ     3$
 7626  033356  104001                          ERROR   .1                      ;CPU ERROR
 7627  033360  022627  033344          3$:     CMP     (SP)+,#1$               ;CHECK THAT STACK CONTAINS CORRECT ADDR.
 7628  033364  001401                          BEQ     4$                      ;
 7629  033366  104001                          ERROR   .1                      ;CPU ERROR
 7630  033370  022627  030000          4$:     CMP     (SP)+,#30000            ;IS THE PSW OK?
 7631  033374  001401                          BEQ     5$                      ;
```

```
                        ************ DOUBLE OPERAND TESTS ******************                                SEQ 0140

 7632  033376  104001                        ERROR   ;1                      ;CPU ERROR
 7633  033400  005067  144362          5$:   CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
 7634  033404  016767  147352  144372        MOV     SLOC00,4                ;RESTORE VECTOR
 7635
 7636  033412                        ODDXX:
 7637
 7638                                ;       ODD ADDRESS/ILLEGAL INST FETCH TRAP TEST
 7639                                ;;**************************************************************
 7640                                ;THIS PROGRAM GENERATES AN ODD ADDRESS IN THE PC. THE KDJ11 SHOULD
 7641                                ;TRAP THROUGH ADDR 4
 7642  033412  005067  144350                CLR     CPEREG                  ;INIT THE CPU ERROR REG
 7643  033416  016767  144362  147336        MOV     4,SLOC00                ;SAVE VECTOR
 7644  033424  012767  033460  144352        MOV     #2$,4                   ;SET UP VECTOR TO HANDLE ODD ADDR TRAP
 7645  033432  016767  144350  147324        MOV     6,SLOC01                ;SAVE VECTOR
 7646  033440  005067  144342                CLR     6                       ;INIT VECTOR
 7647  033444  012746  030000                MOV     #30000,-(SP)            ;PUSH A KNOWN PSW ON THE STACK
 7648  033450  012746  033457                MOV     #1$+1,-(SP)             ;PUSH AN ODD NUMBER ON THE STACK
 7649  033454  000002                        RTI                             ;POP ODD ADDRESS OFF STACK INTO PC
 7650                                                                        ;SHOULD TRAP HERE
 7651  033456  104001                  1$:   ERROR   ;1                      ;CPU ERROR
 7652  033460  022767  000100  144300  2$:   CMP     #BIT06,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
 7653  033466  001401                        BEQ     3$
 7654  033470  104001                        ERROR   ;1                      ;CPU ERROR
 7655  033472  022726  033457          3$:   CMP     #1$+1,(SP)+             ;IS STACK CONTENTS CORRECT?
 7656  033476  001401                        BEQ     4$                      ;
 7657  033500  104001                        ERROR   ;1                      ;CPU ERROR
 7658  033502  022726  030000          4$:   CMP     #30000,(SP)+            ;IS PSW CORRECT ON STACK?
 7659  033506  001401                        BEQ     5$                      ;
 7660  033510  104001                        ERROR   ;1                      ;CPU ERROR
 7661  033512  005067  144250          5$:   CLR     CPEREG                  ;CLEAR CPU ERROR REG
 7662                                ;
 7663                                ;NOW WE'LL TRY TO FETCH AN INSTRUCTION FROM AN INTERNAL REGISTER.
 7664                                ;THIS SHOULD CAUSE A TRAP TO ADDR 4 AND SET BIT 6 IN THE CPU
 7665                                ;ERROR REGISTER.
 7666                                ;
 7667  033516  012767  033544  144260        MOV     #7$,4                   ;LOAD VECTOR WITH TRAP HANDLER ADDR
 7668  033524  012767  030340  144254        MOV     #30340,6                ;LOAD VEC WITH PSW VALUE ON TRAP
 7669  033532  005067  144240                CLR     PS                      ;CLEAR THE PSW
 7670  033536  C00167  144234                JMP     PS                      ;*******TEST INSTRUCTION********
 7671                                                                        ;TRY INSTRUCTION FETCH FROM INTERNAL
 7672                                                                        ;REGISTER-- SHOULD TRAP VIA ADDR 4
 7673  033542  104001                  6$:   ERROR   ;1                      ;CPU ERROR
 7674  033544  016701  144226          7$:   MOV     PS,R1                   ;SAVE CONTENTS OF PSW IN R1
 7675  033550  022767  000100  144210        CMP     #BIT06,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
 7676  033556  001401                        BEQ     8$                      ;
 7677  033560  104001                        ERROR   ;1                      ;CPU ERROR
 7678  033562  022726  177776          8$:   CMP     #PS,(SP)+               ;IS STACK CONTENTS CORRECT?
 7679  033566  001401                        BEQ     9$                      ;
 7680  033570  104001                        ERROR   ;1                      ;CPU ERROR
 7681  033572  022726  000000          9$:   CMP     #0,(SP)+                ;IS STACK CONTENTS CORRECT?
 7682  033576  001401                        BEQ     10$                     ;
 7683  033600  104001                        ERROR   ;1                      ;CPU ERROR
 7684  033602  022701  000340          10$:  CMP     #340,R1                 ;WAS PSW LOADED PROPERLY ON TRAP?
 7685  033606  001401                        BEQ     11$                     ;
 7686  033610  104001                        ERROR   ;1                      ;CPU ERROR
 7687  033612  005067  144150          11$:  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
```

```
 7688  033616  016767  147140  144160          MOV     SLOC00,4                ;RESTORE VECTOR
 7689  033624  016767  147134  144154          MOV     SLOC01,6                ;       "
 7690
 7691  033632                          RXXX:
 7692
 7693                                  ;       RED ZONE TRAP TEST
 7694  033632  013767  000004  147122          MOV     @#4,SLOC00              ;SAVE VECTOR
 7695  033640  012737  033670  000004          MOV     #2$,@#4                 ;SET UP VECTOR
 7696  033646  012706  000777                  MOV     #777,R6                 ;SET UP THE STACK WITH ODD ADDRESS.
 7697  033652  005067  144110                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
 7698  033656  005067  144114                  CLR     PSW                     ;CLEAR THE PSW
 7699  033662  005737  177700                  TST     @#177700                ;ACCESS NON-EXISTANT I/O ADDRESS
 7700  033666  104001                  1$:     ERROR   +1              ;CPU ERROR
 7701  033670  022706  000000          2$:     CMP     #0,SP                   ;IS R6 CORRECT?
 7702  033674  001401                          BEQ     3$                      ;BRANCH IF YES
 7703  033676  104001                          ERROR   +1              ;CPU ERROR
 7704  033700  022726  033666          3$:     CMP     #1$,(SP)+               ;IS DATA AT ADDR 0 CORRECT?
 7705  033704  001401                          BEQ     4$                      ;BRANCH IF YES
 7706  033706  104001                          ERROR   +1              ;CPU ERROR
 7707  033710  022716  000000          4$:     CMP     #0,(SP)                 ;IS PSW DATA IN ADDR 2 CORRECT?
 7708  033714  001401                          BEQ     5$                      ;BRANCH IF YES
 7709  033716  104001                          ERROR   +1              ;CPU ERROR
 7710  033720  022767  000124  144040  5$:     CMP     #124,CPEREG             ;IS CPU ERROR REGISTER CORRECT?
 7711  033726  001401                          BEQ     6$                      ;BRANCH IF YES
 7712  033730  104001                          ERROR   +1              ;CPU ERROR
 7713  033732  005067  144030          6$:     CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7714  033736  012706  001000                  MOV     #STBOT,SP               ;RESTORE STACK
 7715  033742  016737  147014  000004          MOV     SLOC00,@#4              ;RESTORE VECTOR
 7716  033750  005037  000000                  CLR     @#0                     ;RESTORE ADDR 0
 7717  033754  005037  000002                  CLR     @#2                     ;RESTORE ADDR 2
 7718
 7719
 7720
 7721  033760                          PIRXXX:
 7722                                  ;       TEST PIRQ REGISTER RESPONSE
 7723  033760  016767  144020  146774          MOV     4,SLOC00                ;SAVE CONTENTS OF VECTORS
 7724  033766  012767  034014  144010          MOV     #PIRQNX,4               ;SETUP INTERRUPT VECTOR
 7725  033774  005067  143766                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7726  034000  005737  177772                  TST     @#PIRQ                  ;LOOK FOR REPLY FROM PIRQ
 7727  034004  016767  146752  143772          MOV     SLOC00,4                ;RESTORE VECTORS
 7728  034012  000401                          BR      PIRTEX                  ;IF IT RESPONDS, CONTINUE TESTING.
 7729                                                                          ;ERROR! NO RESPONSE FROM PIRQ REG
 7730  034014  104001                  PIRQNX: ERROR   +1              ;CPU ERROR
 7731  034016                          PIRTEX:
 7732
 7733  034016                          PIR1:
 7734                                  ;       TEST PIRQ REGISTER DATA
 7735  034016  013767  000240  146736          MOV     @#PIRQVEC,SLOC00        ;SAVE PIRQ VECTORS
 7736  034024  013767  000242  146732          MOV     @#PIRQVEC+2,SLOC01      ;       "
 7737  034032  012737  034242  000240          MOV     #UNXPIR,@#PIRQVEC       ;SET UP PIRQ VECTOR FOR UNEXPECTED INTERRUPT
 7738  034040  012737  000340  000242          MOV     #PR7,@#PIRQVEC+2        ;
 7739  034046  012703  001000                  MOV     #1000,R3                ;PUT 1000 IN R3: START TESTING
 7740                                                                          ;BITS IN PIRQ REG BY FLOATING
 7741                                                                          ;A BIT THROUGH BITS 9-15.
 7742  034052  012704  034116                  MOV     #PIRTBL,R4              ;SET UP R4 AS A POINTER TO EXPECTED
 7743                                                                          ;ENCODED PRIORITY LEVELS IN PIRTBL
```

```
                        ************ DOUBLE OPERAND TESTS ****************                SEQ 0142

7744  034056  000237                1$:     SPL     7                       ;DON'T ALLOW INTERRUPTS.
7745  034060  005037  177772                CLR     @#PIRQ                  ;CLEAR OUT THE PIRQ
7746  034064  023724  177772                CMP     @#PIRQ,(R4)+            ;IS PIRQ OK??
7747  034070  001401                        BEQ     2$                      ;BRANCH IF OK
7748                                                                        ;ERROR; PIRQ REG WAS NOT CLEAR
7749  034072  104001                        ERROR   +1              ;CPU ERROR
7750  034074  010337  177772        2$:     MOV     R3,@#PIRQ               ;SET A BIT IN PIRQ REGISTER
7751  034100  023724  177772                CMP     @#PIRQ,(R4)+            ;COMPARE THE ENCODED PRIORITY BITS
7752                                                                        ;WITH DATA IN THE TABLE (PIRTBL)
7753  034104  001401                        BEQ     3$                      ;BRANCH IF ITS OK
7754                                                                        ;ERROR; ENCODED PRIORITY LEVELS ARE
7755                                                                        ;NOT CORRECT
7756  034106  104001                        ERROR   +1              ;CPU ERROR
7757  034110  006303        3$:     ASL     R3                      ;FLOAT A "1" THRU BITS 9-15
7758  034112  103370                        BCC     2$                      ;IF CARRY BIT ISN'T SET, DO AGAIN.
7759  034114  000410                        BR      EXPIR1                  ;GO TO EXIT TEST.
7760
7761  034116  000000        PIRTBL: .WORD   0
7762  034120  001042                .WORD   1042
7763  034122  002104                .WORD   2104
7764  034124  004146                .WORD   4146
7765  034126  010210                .WORD   10210
7766  034130  020252                .WORD   20252
7767  034132  040314                .WORD   40314
7768  034134  100356                .WORD   100356
7769
7770  034136                EXPIR1:
7771
7772
7773  034136                PIR2:
7774                                ;       TEST PIRQ REGISTER LEVEL ENCODING
7775                                ;;**************************************************************
7776                                ;THIS TEST IS TO CHECK THAT THE HIGHEST PRIORITY LEVEL SET IN THE
7777                                ;PROGRAM INTERRUPT REQUEST BITS IS REFLECTED IN THE ENCODED PROGRAM
7778                                ;INTERRUPT ACTIVE BITS.
7779  034136  000237                        SPL     7                       ;SHUT OFF INTERRUPTS
7780  034140  005037  177772                CLR     @#PIRQ                  ;CLEAR PIRQ REGISTER
7781  034144  012767  000001  146624        MOV     #1,FLAG                 ;SETUP END OF LOOP SIGNAL
7782  034152  012703  177000                MOV     #177000,R3              ;SET UP DESIRED PATTERN IN R3
7783  034156  012704  034222                MOV     #PITBL1,R4              ;SETUP R4 AS A TABLE POINTER
7784  034162  010337  177772        1$:     MOV     R3,@#PIRQ               ;SET PATERN IN PIRQ REGISTER
7785
7786  034166  023724  177772                CMP     @#PIRQ,(R4)+            ;COMPARE PATTERN IN PIRQ WITH PATTERN IN
7787                                                                        ;EXPECTED PATTERN TABLE.
7788  034172  001012                        BNE     2$                      ;GO TO ERROR IF NOT THE SAME
7789  034174  005737  002776                TST     @#FLAG                  ;IS THIS THE LAST TIME THROUGH??
7790  034200  001421                        BEQ     PIR2EX                  ;YES, IF FLAG IS ZERO
7791  034202  006003                        ROR     R3                      ;SHIFT PATTERN TO RIGHT FOR NEXT TEST
7792  034204  042703  000777                BIC     #777,R3                 ;STRIP OFF BITS  8-0
7793  034210  001364                        BNE     1$                      ;IF R3 IS NOT = 0 GO DO THE NEXT PATTERN
7794  034212  005037  002776                CLR     @#FLAG                  ;CLR THE FLAG TO INDICATE LAST
7795                                                                        ;TIME THROUGH THE LOOP
7796  034216  000761                        BR      1$                      ;GO THROUGH LOOP ONCE MORE
7797                                                                        ;ERROR; PATTERNS WERE NOT THE SAME
7798  034220  104001        2$:     ERROR   +1              ;CPU ERROR
7799
```

```
7800  034222  177356  077314  037252  PITBL1: .WORD   177356,77314,37252,17210,.146,3104,1042,0
7801  034230  017210  007146  003104
7802  034236  001042  000000
7803
7804  034242                          UNXPIR:                                 ;UNEXPECTED PIRQ INTERRUPT
7805  034242  104001                          ERROR   .1              ;CPU ERROR
7806
7807  034244                          PIR2EX:
7808
7809  034244                          PIR3:
7810                                  ;       TEST PIRQ INTERRUPTS
7811                                  ;;**************************************************************
7812                                  ;THIS TEST CHECKS THAT EACH PROGRAM INTERRUPT OCCURS PROPERLY
7813  034244  012703  001000                  MOV     #1000,R3                ;SETUP R3 AS A WORKING REGISTER
7814  034250  012737  034314  000240        MOV     #PIRRTN,@#PIRQVEC       ;SET UP INTERRUPT ROUTINE AT PIR VECTOR
7815  034256  012737  000340  000242        MOV     #340,@#PIRQVEC+2        ;       "       "       "       "
7816  034264  005004                        CLR     R4                      ;INITIALIZE R4 AS EXPECTED DATA HOLDER.
7817  034266  005724                  PI1:    TST     (R4)+                   ;INCREMENT R4 BY 2
7818  034270  050337  177772                BIS     R3,@#PIRQ               ;SET PIRQ TO INTERRUPT
7819  034274  000230                        SPL     0                       ;ENABLE INTERRUPT TO OCCUR
7820                                                                        ;ERROR! INTERRUPT DID NOT OCCUR
7821  034276  104001                        ERROR   .1              ;CPU ERROR
7822                                                                        ;ERROR! INTERRUPT OCCURRED IN WRONG SEQUENCE
7823  034300  104001                  PI2:    ERROR   .1              ;CPU ERROR
7824  034302  062706  000004          PI3:    ADD     #4,SP                   ;CLEAN UP THE STACK
7825  034306  006303                        ASL     R3                      ;SHIFT R3 TO LEFT FOR NEXT INTERRUPT
7826  034310  103366                        BCC     PI1                     ;END IF CARRY BIT IS SET
7827  034312  000412                        BR      PIR3EX                  ;ON TO THE NEXT TEST
7828
7829  034314  013705  177772          PIRRTN: MOV     @#PIRQ,R5               ;MOVE THE CONTENTS OF PIRQ REG TO R5
7830  034320  005037  177772                CLR     @#PIRQ                  ;KILL PIRQ INTERRUPT.
7831  034324  042705  177761                BIC     #177761,R5              ;MASK OFF ALL BITS EXCEPT 1-4.
7832  034330  020405                        CMP     R4,R5                   ;DID THE CORRECT LEVEL INTERRUPT OCCUR?
7833  034332  001362                        BNE     PI2                     ;IF NOT; GO TO ERROR.
7834  034334  000167  177742                JMP     PI3                     ;RETURN FROM SUCCESSFUL INTERRUPT,GET
7835                                                                        ;READY FOR THE NEXT ONE.
7836
7837  034340                          PIR3EX:
7838
7839
7840  034340                          PIR4:
7841                                  ;       TEST PIRQ VS PSW INTERRUPT LEVEL
7842                                  ;;**************************************************************
7843                                  ;THIS TEST IS TO ENSURE THAT PIRQ CANNOT INTERRUPT WHEN PSW INTERRUPT
7844                                  ;LEVEL IS SET TO BLOCK THEM OUT.
7845  034340  005037  177772                  CLR     @#PIRQ                  ;CLEAR THE PIRQ
7846  034344  005037  177776                CLR     @#PSW                   ;CLEAR THE PSW.
7847  034350  000237                        SPL     7                       ;BLOCK INTERRUPTS FROM BEING SERVICED.
7848  034352  012703  001000                MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
7849  034356  012737  034410  000240        MOV     #2$,@#PIRQVEC           ;SETUP INTERRUPT VECTORS
7850  034364  012737  000340  000242        MOV     #340,@#PIRQVEC+2                ;
7851  034372  010337  177772          1$:     MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
7852  034376  006303                        ASL     R3                      ;SHIFT A "1" LEFT TO INCREASE INTERRUPT
7853                                                                        ;PRIORITY LEVEL.
7854  034400  103374                        BCC     1$                      ;IF C BIT NOT SET THEN DO IT AGAIN.
7855  034402  005037  177772                CLR     @#PIRQ                  ;ELSE CLEAR THE PIRQ
```

```
                        ;************ DOUBLE OPERAND TESTS *****************                    SEQ 0144

7856  034406  000403                BR      3$                      ;EXIT TEST.
7857
7858  034410  005037  177772    2$:   CLR     @#PIRQ                  ;STOP PIRQ FROM INTERRUPTING.
7859                                                                  ;ERROR! NO INTERRUPTS SHOULD OCCUR.
7860  034414  104001                ERROR   .1              ;CPU ERROR
7861  034416                    3$:
7862
7863
7864  034416                    PIR5:
7865                            ;       TEST PIRQ INTERRUPTS OCCUR AT PROPER LEVEL
7866                            ;;*****************************************************************
7867                            ;THIS TEST ENSURES THAT INTERRUPTS OCCUR AT THE PROPER LEVEL
7868                            ;THE PRIORITY LEVEL IS INITIALLY SET TO PR76. THE PIRQ THEN STARTS INTERRUPTING
7869                            ;AT PRI1. NO INTERRUPTS SHOULD OCCUR UNTIL THE INTERRUPTING LEVEL EXCEEDS THE
7870                            ;PRIORITY LEVEL IN THE PSW. AFTER EACH LEVEL OF INTERRUPT HAS BEEN TRIED; THE
7871                            ;PSW PRIORITY LEVEL IS LOWERED ONE NOTCH, AND THE CYCLE REPEATS UNTIL PSW
7872                            ;PRIORITY LEVEL 0 IS REACHED. INTERRUPTS AT PSW PRIORITY LEVEL 0 HAVE BEEN DONE
7873                            ;PREVIOUSLY.
7874  034416  005037  177772          CLR     @#PIRQ                  ;CLR THE PIRQ
7875  034422  012737  034534  000240  MOV     #10$,@#PIRQVEC          ;SET UP INTERRUPT VECTORS
7876  034430  012737  000340  000242  MOV     #PR7,@#PIRQVEC+2
7877  034436  012705  034612          MOV     #PITBL2,R5              ;SETUP R5 AS A TABLE POINTER.
7878  034442  012737  000006  110146  MOV     #6,@#DCOUNT             ;INIT LOCATION DCOUNT:
7879  034450  005004                1$:   CLR     R4                      ;INITIALIZE R4 AS A COUNTER.
7880  034452  012703  001000          MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
7881  034456  012567  143314          MOV     (R5)+,PS                ;MOVE PRIORITY LEVEL TO PSW
7882  034462  005724                3$:   TST     (R4)+                   ;INCREMENT R4 BY 2
7883  034464  010337  177772          MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
7884                            ;*******INTERRUPTS WILL HAPPEN HERE*********
7885  034470  013701  177776          MOV     @#PS,R1                 ;SAVE PS IN R1          !ONLY EXECUTED
7886  034474  013702  177772          MOV     @#PIRQ,R2               ;SAVE PIRQ IN R2        !   IF NO
7887  034500  042701  177437          BIC     #177437,R1              ;CLEAR EXTRANEOUS BITS  ! INTERRUPT
7888  034504  042702  177437          BIC     #177437,R2              ;                       !    HAS
7889  034510  020102                CMP     R1,R2                   ;R1 SHOULD BE >= R2     !  OCCURRED
7890  034512  002001                BGE     4$                      ;IF IT IS GO ON         !-------
7891                                                                  ;ERROR! SHOULD HAVE INTERRUPTED.
7892  034514  104001                ERROR   .1              ;CPU ERROR
7893  034516  006303            4$:   ASL     R3                      ;SHIFT A "1" LEFT UNTIL INTERRUPT LEVEL
7894                                                                  ;IS REACHED.
7895  034520  103360                BCC     3$                      ;BRANCH BACK IF CARRY IS NOT SET.
7896  034522  005337  110146          DEC     @#DCOUNT                ;LOWER INTERRUPT PRIORITY LEVEL
7897  034526  001350                BNE     1$                      ;IF DCOUNT= 0 , WE'RE DONE.
7898  034530  000167  000072          JMP     PIRSEX                  ;JUMP TO END OF THIS TEST.
7899
7900                            ;PIRQ INTERRUPT SERVICE ROUTINE
7901                            ;
7902
7903  034534  013746  177772    10$:  MOV     @#PIRQ,-(SP)            ;SAVE PIRQ DATA ON STACK
7904  034540  005037  177772          CLR     @#PIRQ                  ;SHUT OFF PIRQ INTERRUPTS
7905  034544  016601  000004          MOV     4(SP),R1                ;GET OLD PSW. SAVE IN R1.
7906  034550  011602                MOV     (SP),R2                 ;GET PIRQ FROM STACK.
7907  034552  042702  177437          BIC     #177437,R2              ;CLEAR UNWANTED BITS FROM R2
7908  034556  042701  177437          BIC     #177437,R1              ;CLEAR UNWANTED BITS FROM R1
7909  034562  020102                CMP     R1,R2                   ;R2 SHOULD BE > R1.
7910  034564  100401                BMI     20$                     ;GO CHECK SEQUENCE OF INTERRUPT.
7911                                                                  ;ERROR! PRIORITY OF INTERRUPT WAS NOT
```

```
                                                        ;HIGH ENOUGH. SHOULDN'T HAVE OCCURRED.
034566  104001                  ERROR   ^1              ;CPU ERROR
034570  012602          20$:    MOV     (SP)+,R2        ;POP OLD PIRQ OFF THE STACK.
034572  042702  177761          BIC     #177761,R2      ;CLEAR OFF EXTRANEOUS BITS.
034576  020402                  CMP     R4,R2           ;SHOULD BE EQUAL.
034600  001401                  BEQ     21$             ;IF THEY ARE EQUAL, CLEAN UP STACK AND
                                                        ;GET READY FOR THE NEXT INTERRUPT
                                                        ;ELSE
                                                        ;ERROR! INTERRUPT OCCURRED OUT OF SEQUENCE.
034602  104001                  ERROR   ^1              ;CPU ERROR
034604  012716  034516  21$:    MOV     #4$,(SP)        ;PUT RETURN ADDRESS ON THE STACK.
034610  000002                  RTI                     ;RETURN FROM INTERRUPT. RESTORE PSW.

034612  000300          PITBL2: .WORD   300             ;PRIORITY LEVEL 6
034614  000240                  .WORD   240             ;PRIORITY LEVEL 5
034616  000200                  .WORD   200             ;PRIORITY LEVEL 4
034620  000140                  .WORD   140             ;PRIORITY LEVEL 3
034622  000100                  .WORD   100             ;PRIORITY LEVEL 2
034624  000040                  .WORD   40              ;PRIORITY LEVEL 1

034626          PIRSEX:

034626          PIR6:
                ;       TEST THAT PIRQS ARE SERVICED IN CORRECT ORDER
                ;;*************************************************************
                ;THIS TEST CHECKS THAT ALL PIRQ INTERRUPTS ARE SERVICED
                ;IN THE CORRECT DECENDING ORDER. I.E. IRQ6 IS NOT SERVICED
                ;BEFORE IRQ7 ETC. THE PIRQ IS LOADED WITH ALL INTERRUPTS SIMULTANEOUSLY
                ;TURNED ON.  THE PSW PRIORITY LEVEL IS THEN LOWERED TO 0. EACH INTERRUPT
                ;SHOULD BE SERVICED IN DECENDING ORDER. EACH TIME AN INTERRUPT OCCURRS, THE PSW
                ;IS LOADED WITH PRIORITY LEVEL 7 WHICH STOPS THE PIRQ FROM FURTHER INTERRUPTS.
                ;AFTER A CORRECT INTERRUPT, AN RTI IS EXECUTED, AND THE PSW PRIORITY LEVEL
                ;RETRUNS TO ZERO ALLOWING THE NEXT LOWER INTERRUPT TO OCCUR.
034626  005037  177772          CLR     @#PIRQ          ;CLEAR OUT THE PIRQ
034632  000237                  SPL     7               ;NO INTERRUPTS WILL BE SERVICED.
034634  012737  034674  000240  MOV     #10$,@#PIRQVEC  ;SET UP VECTORS
034642  012737  000340  000242  MOV     #PR7,@#PIRQVEC+2;
034650  012703  177000          MOV     #177000,R3      ;SETUP DESIRED PATTERN IN R3
034654  010337  177772          MOV     R3,@#PIRQ       ;MOVE PATTERN IN R3 TO PIRQ.
034660  012704  000016          MOV     #16.,R4         ;PRELOAD R4 AS A DECREMENTING COUNTER.
034664  000230                  SPL     0               ;LET THE PIRQ INTERRUPT
                ;*********INTERRUPTS OCCUR HERE!!!*******************
034666  005704                  TST     R4              ;IS R4 ZERO? IT SHOULD BE. THIS
                                                        ;INSTRUCTION SHOULDN'T BE EXECUTED
                                                        ;UNTIL ALL INTERRUPTS HAVE OCCURRED.
034670  001417                  BEQ     PIR6EX          ;GO TO NEXT TEST.
                                                        ;ERROR! ALL INTERRUPTS DID NOT OCCUR.
034672  104001                  ERROR   ^1              ;CPU ERROR
034674  013702  177772  10$:    MOV     @#PIRQ,R2       ;SAVE THE PIRQ IN R2
034700  042702  177761          BIC     #177761,R2      ;STRIP OFF EXTRANEOUS BITS.
034704  020402                  CMP     R4,R2           ;SHOULD BE EQUAL.
034706  001401                  BEQ     15$             ;SETUP FOR NEXT INTERRUPT.
                                                        ;ERROR! INTERRUPT WAS SERVICED OUT OF ORDER
034710  104001                  ERROR   ^1              ;CPU ERROR
034712  005744          15$:    TST     -(R4)           ;DECREMENT R4 BY 2
```

```
 7968  034714  006003                          ROR     R3                      ;CHANGE PATTERN IN R3; LOWER INTERRUPT PRIORITY
 7969  034716  042703  000777                  BIC     #777,R3                 ;STRIP OFF UNNEEDED BITS
 7970  034722  010337  177772                  MOV     R3,@#PIRQ               ;LOWER INTERRUPT PRIORITY LEVEL.
 7971  034726  000002                          RTI                             ;
 7972
 7973  034730  016737  146026  000240  PIR6EX: MOV     SLOC00,@#PIRQVEC        ;RESTORE VECTORS
 7974  034736  016737  146022  000242          MOV     SLOC01,@#PIRQVEC+2              ;       "
 7975                                                                  ;
 7976
 7977                                  .SBTTL  MEMORY MANAGEMENT TESTS
 7978                                  ;*******************************************************************************
 7979                                  ;*******************************************************************************
 7980                                  ;                       BEGIN MMU TESTING
 7981                                  ;*******************************************************************************
 7982                                  ;*******************************************************************************
 7983  034744                          TSMMU1:
 7984                                  ;       STATUS REGISTER TEST
 7985  034744  005067  143016                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7986  034750  005037  1775 2                  CLR     @#177572                ;TURN MMU OFF
 7987  034754  005037  002776                  CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
 7988  034760  013746  000004                  MOV     @#4,-(SP)               ;SAVE OLD VECTOR
 7989  034764  012737  132512  000004          MOV     #ADDTRP,@#4             ;SETUP NEW VECTOR
 7990  034772  005005                          CLR     R5                      ;CLEAR FLAG
 7991  034774  013701  177572                  MOV     @#177572,R1             ; TEST MMR0
 7992  035000  013701  177574                  MOV     @#177574,R1             ; TEST MMR1
 7993  035004  013701  177576                  MOV     @#177576,R1             ; TEST MMR2
 7994  035010  013701  172516                  MOV     @#172516,R1             ; TEST MMR3
 7995  035014  012637  000004                  MOV     (SP)+,@#4               ;RESTORE VECTOR
 7996  035020  020527  000000                  CMP     R5,#0                   ;DID WE TRAP
 7997  035024  001401                          BEQ     1$                      ;NO, THEN BRANCH
 7998  035026  104002                          ERROR   +2                      ;MMU ERROR
 7999                                                                          ;YES, GO TO ERROR
 8000  035030                          1$:
 8001  035030                          TSMMU2:
 8002                                  ;       ADDRESS TEST OF PARS,PDRS, AND FP REGS
 8003  035030  005067  142732                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 8004  035034  005037  177572                  CLR     @#177572                ;MMU OFF
 8005  035040  005037  002776                  CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
 8006  035044  013746  000244                  MOV     @#244,-(SP)             ;SAVE FP VECTOR
 8007  035050  013746  000246                  MOV     @#246,-(SP)
 8008  035054  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
 8009  035060  012737  000246  000244          MOV     #246,@#244              ;SETUP NEW FP VECTOR
 8010  035066  012737  000002  000246          MOV     #2,@#246                ;
 8011  035074  012737  132512  000004          MOV     #ADDTRP,@#4             ;SETUP NEW TIME OUT VECTOR
 8012  035102  005005                          CLR     R5                      ;CLEAR TIMEOUT FLAG
 8013  035104  012700  172200                  MOV     #172200,R0              ;LOAD ALL PARS AND PDRS WITH ZERO
 8014  035110  005020                  1$:     CLR     (R0)+                   ;
 8015  035112  020027  172400                  CMP     R0,#172400              ;
 8016  035116  001374                          BNE     1$                      ;
 8017  035120  012700  177600                  MOV     #177600,R0              ;
 8018  035124  005020                  2$:     CLR     (R0)+                   ;
 8019  035126  020027  177700                  CMP     R0,#177700              ;
 8020  035132  001374                          BNE     2$                      ;
 8021  035134  170127  000200                  LDFPS   #200                    ;
 8022  035140  012700  003012                  MOV     #FLOAT,R0               ;LOAD AC0-AC5 WITH 0
 8023  035144  005020                          CLR     (R0)+                   ;
```

```
8024  035146  005020                      CLR     (R0)+                   ;
8025  035150  005020                      CLR     (R0)+                   ;
8026  035152  005020                      CLR     (R0)+                   ;
8027  035154  012700  003012              MOV     #FLOAT,R0               ;
8028  035160  172410                      LDD     (R0),AC0                ;
8029  035162  172510                      LDD     (R0),AC1                ;
8030  035164  172610                      LDD     (R0),AC2                ;
8031  035166  172710                      LDD     (R0),AC3                ;
8032  035170  174004                      STD     AC0,AC4                 ;
8033  035172  174005                      STD     AC0,AC5                 ;
8034  035174  174500              3$:     DIVD    AC0,AC1                 ;LOAD FEC WITH 4 AND FEA WITH #3$
8035  035176  170337  003022              STST    @#FLO                   ;CHECK FEC FOR 4 AND FEA FOR #3$
8036  035202  012704  003022              MOV     #FLO,R4                 ;
8037  035206  022427  000004              CMP     (R4)+,#4                ;
8038  035212  001401                      BEQ     21$                     ;
8039  035214  104002                      ERROR   ^2              ;MMU ERROR
8040                                                                      ;
8041  035216  021427  035174      21$:    CMP     (R4),#3$                ;
8042  035222  001401                      BEQ     22$                     ;
8043  035224  104002                      ERROR   ^2              ;MMU ERROR
8044                                                                      ;
8045  035226  012704  172200      22$:    MOV     #172200,R4              ;CHECK EACH PAR, PDR FOR 0 THEN
8046  035232  012701  000001              MOV     #1,R1                   ;WRITE A UNIQUE NUMBER TO IT
8047  035236  010102              4$:     MOV     R1,R2                   ;
8048  035240  072227  000010              ASH     #10,R2                  ;
8049  035244  021427  000000              CMP     (R4),#0                 ;
8050  035250  001401                      BEQ     5$                      ;
8051  035252  104002                      ERROR   ^2              ;MMU ERROR
8052                                                                      ;
8053  035254  010224              5'      MOV     R2,(R4)+                ;
8054  035256  005201                      INC     R1                      ;
8055  035260  020427  172400              CMP     R4,#172400              ;
8056  035264  001364                      BNE     4$                      ;
8057  035266  012704  177600              MOV     #177600,R4              ;
8058  035272  010102              6$:     MOV     R1,R2                   ;
8059  035274  072227  000010              ASH     #10,R2                  ;
8060  035300  021427  000000              CMP     (R4),#0                 ;
8061  035304  001401                      BEQ     7$                      ;
8062  035306  104002                      ERROR   ^2              ;MMU ERROR
8063                                                                      ;
8064  035310  010224              7$:     MOV     R2,(R4)+                ;
8065  035312  005201                      INC     R1                      ;
8066  035314  020427  177700              CMP     R4,#177700              ;
8067  035320  001364                      BNE     6$                      ;
8068  035322  012704  003022              MOV     #FLO,R4                 ;CHECK AC5 FOR ALL ZEROES THEN LOAD A 6
8069  035326  012703  003012              MOV     #FLOAT,R3               ;
8070  035332  174014                      STD     AC0,(R4)                ;
8071  035334  172405                      LDD     AC5,AC0                 ;
8072  035336  174013                      STD     AC0,(R3)                ;
8073  035340  012702  000004              MOV     #4,R2                   ;
8074  035344  022327  000000      8$:     CMP     (R3)+,#0                ;
8075  035350  001401                      BEQ     9$                      ;
8076  035352  104002                      ERROR   ^2              ;MMU ERROR
8077                                                                      ;
8078  035354  005302              9$:     DEC     R2                      ;
8079  035356  001372                      BNE     8$                      ;
```

```
8080  035360  012703  003012                  MOV     #FLOAT,R3               ;
8081  035364  012713  000006                  MOV     #6,(R3)                 ;
8082  035370  172413                          LDD     (R3),AC0                ;
8083  035372  174005                          STD     AC0,AC5                 ;
8084  035374  172404                          LDD     AC4,AC0                 ;CHECK AC4 FOR ALL ZEROES THEN LOAD A 5
8085  035376  174013                          STD     AC0,(R3)                ;
8086  035400  012702  000004                  MOV     #4,R2                   ;
8087  035404  022327  000000          10$:    CMP     (R3)+,#0                ;
8088  035410  001401                          BEQ     11$                     ;
8089  035412  104002                          ERROR   .2              ;MMU ERROR
8090                                                                  ;
8091  035414  005302                  11$:    DEC     R2                      ;
8092  035416  001372                          BNE     10$                     ;
8093  035420  012703  003012                  MOV     #FLOAT,R3               ;
8094  035424  012713  000005                  MOV     #5,(R3)                 ;
8095  035430  172413                          LDD     (R3),AC0                ;
8096  035432  174004                          STD     AC0,AC4                 ;
8097  035434  012702  000004                  MOV     #4,R2                   ;CHECK AC0 FOR ALL ZEROES THEN LOAD A 1
8098  035440  022427  000000          12$:    CMP     (R4)+,#0                ;
8099  035444  001401                          BEQ     13$                     ;
8100  035446  104002                          ERROR   .2              ;MMU ERROR
8101                                                                  ;
8102  035450  005302                  13$:    DEC     R2                      ;
8103  035452  001372                          BNE     12$                     ;
8104  035454  012713  000001                  MOV     #1,(R3)                 ;
8105  035460  172413                          LDD     (R3),AC0                ;
8106  035462  012704  003022                  MOV     #FLO,R4                 ;CHECK AC1 FOR ALL ZEROES THEN LOAD A 2
8107  035466  012702  000004                  MOV     #4,R2                   ;
8108  035472  174114                          STD     AC1,(R4)                ;
8109  035474  022427  000000          14$:    CMP     (R4)+,#0                ;
8110  035500  001401                          BEQ     15$                     ;
8111  035502  104002                          ERROR   .2              ;MMU ERROR
8112                                                                  ;
8113  035504  005302                  15$:    DEC     R2                      ;
8114  035506  001372                          BNE     14$                     ;
8115  035510  012713  000002                  MOV     #2,(R3)                 ;
8116  035514  172513                          LDD     (R3),AC1                ;
8117  035516  012704  003022                  MOV     #FLO,R4                 ;CHECK AC2 FOR ALL ZEROES THEN LOAD A 3
8118  035522  012702  000004                  MOV     #4,R2                   ;
8119  035526  174214                          STD     AC2,(R4)                ;
8120  035530  022427  000000          16$:    CMP     (R4)+,#0                ;
8121  035534  001401                          BEQ     17$                     ;
8122  035536  104002                          ERROR   .2              ;MMU ERROR
8123                                                                  ;
8124  035540  005302                  17$:    DEC     R2                      ;
8125  035542  001372                          BNE     16$                     ;
8126  035544  012713  000003                  MOV     #3,(R3)                 ;
8127  035550  172613                          LDD     (R3),AC2                ;
8128  035552  012704  003022                  MOV     #FLO,R4                 ;CHECK AC3 FOR ALL ZEROES THEN LOAD A 4
8129  035556  012702  000004                  MOV     #4,R2                   ;
8130  035562  174314                          STD     AC3,(R4)                ;
8131  035564  022427  000000          18$:    CMP     (R4)+,#0                ;
8132  035570  001401                          BEQ     19$                     ;
8133  035572  104002                          ERROR   .2              ;MMU ERROR
8134                                                                  ;
8135  035574  005302                  19$:    DEC     R2                      ;
```

```
 8136  035576  001372                        BNE     18$                     ;
 8137  035600  012713  000004                MOV     #4,(R3)                 ;
 8138  035604  172713                        LDD     (R3),AC3                ;
 8139  035606  012704  003022                MOV     #FLO,R4                 ;CHECK FPS FOR 100204 THEN LOAD IT WITH 200
 8140  035612  170214                        STFPS   (R4)                    ;
 8141  035614  022714  100204                CMP     #100204,(R4)            ;
 8142  035620  001401                        BEQ     20$                     ;
 8143  035622  104002                        ERROR   +2              ;MMU ERROR
 8144                                                                  ;
 8145  035624  170127  000200        20$:    LDFPS   #200                    ;
 8146  035630  012704  172200                MOV     #172200,R4              ;CHECK PDR, PAR FOR UNIQUE NUMBERS
 8147  035634  012701  000001                MOV     #1,R1                   ;
 8148  035640  010102                23$:    MOV     R1,R2                   ;
 8149  035642  072227  000010                ASH     #10,R2                  ;
 8150  035646  022402                        CMP     (R4)+,R2                ;
 8151  035650  001401                        BEQ     24$                     ;
 8152  035652  104002                        ERROR   +2              ;MMU ERROR
 8153                                                                  ;
 8154  035654  005201                24$:    INC     R1                      ;
 8155  035656  020427  172400                CMP     R4,#172400              ;
 8156  035662  001366                        BNE     23$                     ;
 8157  035664  012704  177600                MOV     #177600,R4              ;
 8158  035670  010102                25$:    MOV     R1,R2                   ;
 8159  035672  072227  000010                ASH     #10,R2                  ;
 8160  035676  022402                        CMP     (R4)+,R2                ;
 8161  035700  001401                        BEQ     26$                     ;
 8162  035702  104002                        ERROR   +2              ;MMU ERROR
 8163                                                                  ;
 8164  035704  005201                26$:    INC     R1                      ;
 8165  035706  020427  177700                CMP     R4,#177700              ;
 8166  035712  001366                        BNE     25$                     ;
 8167  035714  012701  003012                MOV     #FLOAT,R1               ;CHECK AC5 FOR #6
 8168  035720  012704  003022                MOV     #FLO,R4                 ;
 8169  035724  174014                        STD     AC0,(R4)                ;
 8170  035726  172405                        LDD     AC5,AC0                 ;
 8171  035730  174011                        STD     AC0,(R1)                ;
 8172  035732  022127  000006                CMP     (R1)+,#6                ;
 8173  035736  001401                        BEQ     27$                     ;
 8174  035740  104002                        ERROR   +2              ;MMU ERROR
 8175                                                                  ;
 8176  035742  012703  000003        27$:    MOV     #3,R3                   ;
 8177  035746  022127  000000        28$:    CMP     (R1)+,#0                ;
 8178  035752  001401                        BEQ     29$                     ;
 8179  035754  104002                        ERROR   +2              ;MMU ERROR
 8180                                                                  ;
 8181  035756  005303                29$:    DEC     R3                      ;
 8182  035760  001372                        BNE     28$                     ;
 8183  035762  012701  003012                MOV     #FLOAT,R1               ;CHECK AC4 FOR #5
 8184  035766  172404                        LDD     AC4,AC0                 ;
 8185  035770  174011                        STD     AC0,(R )                ;
 8186  035772  022127  000005                CMP     (R1)+,#5                ;
 8187  035776  001401                        BEQ     30$                     ;
 8188  036000  104002                        ERROR   +2              ;MMU ERROR
 8189                                                                  ;
 8190  036002  012703  000003        30$:    MOV     #3,R3                   ;
 8191  036006  022127  000000        31$:    CMP     (R1)+,#0                ;
```

```
 8192  036012  001401                        BEQ     32$                     ;
 8193  036014  104002                        ERROR   .2                     ;MMU ERROR
 8194                                                                        ;
 8195  036016  005303              32$:      DEC     R3                              ;
 8196  036020  001372                        BNE     31$                             ;
 8197  036022  022427  000001                CMP     (R4)+,#1                        ;CHECK AC0 FOR #1
 8198  036026  001401                        BEQ     33$                             ;
 8199  036030  104002                        ERROR   .2                      ;MMU ERROR
 8200                                                                        ;
 8201  036032  012703  000003        33$:    MOV     #3,R3                           ;
 8202  036036  022427  000000        34$:    CMP     (R4)+,#0                        ;
 8203  036042  001401                        BEQ     35$                             ;
 8204  036044  104002                        ERROR   .2                      ;MMU ERROR
 8205                                                                        ;
 8206  036046  005303              35$:      DEC     R3                              ;
 8207  036050  001372                        BNE     34$                             ;
 8208  036052  012701  003012                MOV     #FLOAT,R1                       ;CHECK AC1 FOR #2
 8209  036056  174111                        STD     AC1,(R1)                        ;
 8210  036060  022127  000002                CMP     (R1)+,#2                        ;
 8211  036064  001401                        BEQ     36$                             ;
 8212  036066  104002                        ERROR   .2                      ;MMU ERROR
 8213                                                                        ;
 8214  036070  012703  000003        36$:    MOV     #3,R3                           ;
 8215  036074  022127  000000        37$:    CMP     (R1)+,#0                        ;
 8216  036100  001401                        BEQ     38$                             ;
 8217  036102  104002                        ERROR   .2                      ;MMU ERROR
 8218                                                                        ;
 8219  036104  005303              38$:      DEC     R3                              ;
 8220  036106  001372                        BNE     37$                             ;
 8221  036110  012701  003012                MOV     #FLOAT,R1                       ;CHECK AC2 FOR #3
 8222  036114  174211                        STD     AC2,(R1)                        ;
 8223  036116  022127  000003                CMP     (R1)+,#3                        ;
 8224  036122  001401                        BEQ     39$                             ;
 8225  036124  104002                        ERROR   .2                      ;MMU ERROR
 8226                                                                        ;
 8227  036126  012703  000003        39$:    MOV     #3,R3                           ;
 8228  036132  022127  000000        40$:    CMP     (R1)+,#0                        ;
 8229  036136  001401                        BEQ     41$                             ;
 8230  036140  104002                        ERROR   .2                      ;MMU ERROR
 8231                                                                        ;
 8232  036142  005303              41$:      DEC     R3                              ;
 8233  036144  001372                        BNE     40$                             ;
 8234  036146  012701  003012                MOV     #FLOAT,R1                       ;CHECK AC3 FOR #4
 8235  036152  174311                        STD     AC3,(R1)                        ;
 8236  036154  022127  000004                CMP     (R1)+,#4                        ;
 8237  036160  001401                        BEQ     42$                             ;
 8238  036162  104002                        ERROR   .2                      ;MMU ERROR
 8239                                                                        ;
 8240  036164  012703  000003        42$:    MOV     #3,R3                           ;
 8241  036170  022127  000000        43$:    CMP     (R1)+,#0                        ;
 8242  036174  001401                        BEQ     44$                             ;
 8243  036176  104002                        ERROR   .2                      ;MMU ERROR
 8244                                                                        ;
 8245  036200  005303              44$:      DEC     R3                              ;
 8246  036202  001372                        BNE     43$                             ;
 8247  036204  020527  000000                CMP     R5,#0                           ;IS TIME OUT FLAG 0
```

```
 8248  036210  001401                        BEQ     45$                     ;YES GO ON
 8249  036212  104002                        ERROR   .2                  ;MMU ERROR
 8250                                                                    ;NO GO TO ERROR
 8251  036214  012637  000004        45$:    MOV     (SP)+,@#4               ;RESTORE TIME OUT VECTOR
 8252  036220  012637  000246              MOV     (SP)+,@#246             ;RESTORE FP VECTOR
 8253  036224  012637  000244              MOV     (SP)+,@#244             ;
 8254
 8255  036230                        TSMMU3:
 8256                                ;       WRITE ALL PARS/PDRS WITH ONES THEN ZEROS
 8257  036230  005037  177572              CLR     @#177572                ;MMU OFF
 8258  036234  005037  002776              CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
 8259  036240  012703  172200              MOV     #172200, 3              ;LOAD ALL PARS AND PDRS WITH ONES
 8260  036244  012723  177777        1$:     MOV     #177777,(R3)+           ;
 8261  036250  020327  172400              CMP     R3,#172400              ;
 8262  036254  001373                        BNE     1$                      ;
 8263  036256  012703  177600              MOV     #177600,R3              ;
 8264  036262  012723  177777        2$:     MOV     #177777,(R3)+           ;
 8265  036266  020327  177700              CMP     R3,#177700              ;
 8266  036272  001373                        BNE     2$                      ;
 8267  036274  012703  172200              MOV     #172200,R3              ;CHECK SPDRS FOR ONES
 8268  036300  022327  177416        3$:     CMP     (R3)+,#177416           ;
 8269  036304  001401                        BEQ     4$                      ;
 8270  036306  104002                        ERROR   .2                  ;MMU ERROR
 8271                                                                    ;
 8272  036310  020327  172240        4$:     CMP     R3,#172240              ;
 8273  036314  001371                        BNE     3$                      ;
 8274  036316  022327  177777        5$:     CMP     (R3)+,#177777           ;CHECK SPARS FOR ONES
 8275  036322  001401                        BEQ     6$                      ;
 8276  036324  104002                        ERROR   .2                  ;MMU ERROR
 8277                                                                    ;
 8278  036326  020327  172300        6$:     CMP     R3,#172300              ;
 8279  036332  001371                        BNE     5$                      ;
 8280  036334  022327  177416        7$:     CMP     (R3)+,#177416           ;CHECK KPDRS FOR ONES
 8281  036340  001401                        BEQ     8$                      ;
 8282  036342  104002                        ERROR   .2                  ;MMU ERROR
 8283                                                                    ;
 8284  036344  020327  172340        8$:     CMP     R3,#172340              ;
 8285  036350  001371                        BNE     7$                      ;
 8286  036352  022327  177777        9$:     CMP     (R3)+,#177777           ;CHECK KPARS FOR ONES
 8287  036356  001401                        BEQ     10$                     ;
 8288  036360  104002                        ERROR   .2                  ;MMU ERROR
 8289                                                                    ;
 8290  036362  020327  172400        10$:    CMP     R3,#172400              ;
 8291  036366  001371                        BNE     9$                      ;
 8292  036370  012703  177600              MOV     #177600,R3              ;CHECK UPDRS FOR ONES
 8293  036374  022327  177416        11$:    CMP     (R3)+,#177416           ;
 8294  036400  001401                        BEQ     12$                     ;
 8295  036402  104002                        ERROR   .2                  ;MMU ERROR
 8296                                                                    ;
 8297  036404  020327  177640        12$:    CMP     R3,#177640              ;
 8298  036410  001371                        BNE     11$                     ;
 8299  036412  022327  177777        13$:    CMP     (R3)+,#177777           ;CHECK UPARS FOR ONES
 8300  036416  001401                        BEQ     14$                     ;
 8301  036420  104002                        ERROR   .2                  ;MMU ERROR
 8302                                                                    ;
 8303  036422  020327  177700        14$:    CMP     R3,#177700              ;
```

```
 8304  036426  001371                  BNE     13$                     ;
 8305  036430  012703  172200          MOV     #172200,R3              ;LOAD ALL PARS AND PDRS WITH ZEROES
 8306  036434  012723  000000  15$:    MOV     #0,(R3)+                ;
 8307  036440  020327  172400          CMP     R3,#172400              ;
 8308  036444  001373                  BNE     15$                     ;
 8309  036446  012703  177600          MOV     #177600,R3              ;
 8310  036452  012723  000000  16$:    MOV     #0,(R3)+                ;
 8311  036456  020327  177700          CMP     R3,#177700              ;
 8312  036462  001373                  BNE     16$                     ;
 8313  036464  012703  172200          MOV     #172200,R3              ;CHECK ALL PARS AND PDRS FOR ZEROES
 8314  036470  022327  000000  17$:    CMP     (R3)+,#0                ;
 8315  036474  001401                  BEQ     18$                     ;
 8316  036476  104002                  ERROR   +2              ;MMU ERROR
 8317                                                          ;
 8318  036500  020327  172400  18$:    CMP     R3,#172400              ;
 8319  036504  001371                  BNE     17$                     ;
 8320  036506  012703  177600          MOV     #177600,R3              ;
 8321  036512  022327  000000  19$:    CMP     (R3)+,#0                ;
 8322  036516  001401                  BEQ     20$                     ;
 8323  036520  104002                  ERROR   +2              ;MMU ERROR
 8324                                                          ;
 8325  036522  020327  177700  20$:    CMP     R3,#177700              ;
 8326  036526  001371                  BNE     19$                     ;
 8327
 8328  036530                  TSMMU4:
 8329                          ;       TEST FOR ADJACENT SHORTS IN PARS/PDRS
 8330  036530  005037  177572          CLR     @#177572                ;MMU OFF
 8331  036534  005067  144236          CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 8332  036540  012700  172200          MOV     #172200,R0              ;LOAD SPDRS WITH ALTERNATING PATTERN
 8333  036544  012720  052404  1$:     MOV     #52404,(R0)+            ;
 8334  036550  012720  125012          MOV     #125012,(R0)+           ;
 8335  036554  020027  172240          CMP     R0,#172240              ;
 8336  036560  001371                  BNE     1$                      ;
 8337  036562  012720  125252  2$:     MOV     #125252,(R0)+           ;LOAD SPARS WITH ALTERNATING PATTERN
 8338  036566  012720  052525          MOV     #52525,(R0)+            ;
 8339  036572  020027  172300          CMP     R0,#172300              ;
 8340  036576  001371                  BNE     2$                      ;
 8341  036600  012720  052404  3$:     MOV     #52404,(R0)+            ;LOAD KPDRS WITH ALTERNATING PATTERN
 8342  036604  012720  125012          MOV     #125012,(R0)+           ;
 8343  036610  020027  172340          CMP     R0,#172340              ;
 8344  036614  001371                  BNE     3$                      ;
 8345  036616  012720  125252  4$:     MOV     #125252,(R0)+           ;LOAD KPARS WITH ALTERNATING PATTERN
 8346  036622  012720  052525          MOV     #52525,(R0)+            ;
 8347  036626  020027  172400          CMP     R0,#172400              ;
 8348  036632  001371                  BNE     4$                      ;
 8349  036634  012700  177600          MOV     #177600,R0              ;LOAD UPDRS WITH ALTERNATING PATTERN
 8350  036640  012720  052404  5$:     MOV     #52404,(R0)+            ;
 8351  036644  012720  125012          MOV     #125012,(R0)+           ;
 8352  036650  020027  177640          CMP     R0,#177640              ;
 8353  036654  001371                  BNE     5$                      ;
 8354  036656  012720  125252  6$:     MOV     #125252,(R0)+           ;LOAD UPARS WITH ALTERNATING PATTERN
 8355  036662  012720  052525          MOV     #52525,(R0)+            ;
 8356  036666  020027  177700          CMP     R0,#177700              ;
 8357  036672  001371                  BNE     6$                      ;
 8358                          ;
 8359  036674  012703  172200          MOV     #172200,R3              ;CHECK SPDRS
```

```
8360  036700  022327  052404          7$:     CMP     (R3)+,#52404            ;
8361  036704  001401                          BEQ     8$                      ;
8362  036706  104002                          ERROR   .2              ;MMU ERROR
8363                                                                  ;
8364  036710  022327  125012          8$:     CMP     (R3)+,#125012           ;
8365  036714  001401                          BEQ     9$                      ;
8366  036716  104002                          ERROR   .2              ;MMU ERROR
8367                                                                  ;
8368  036720  020327  172240          9$:     CMP     R3,#172240              ;
8369  036724  001365                          BNE     7$                      ;
8370  036726  022327  125252          10$:    CMP     (R3)+,#125252           ;CHECK SPARS
8371  036732  001401                          BEQ     11$                     ;
8372  036734  104002                          ERROR   .2              ;MMU ERROR
8373                                                                  ;
8374  036736  022327  052525          11$:    CMP     (R3)+,#52525            ;
8375  036742  001401                          BEQ     12$                     ;
8376  036744  104002                          ERROR   .2              ;MMU ERROR
8377                                                                  ;
8378  036746  020327  172300          12$:    CMP     R3,#172300              ;
8379  036752  001365                          BNE     10$                     ;
8380  036754  022327  052404          13$:    CMP     (R3)+,#52404            ;CHECK KPDRS
8381  036760  001401                          BEQ     14$                     ;
8382  036762  104002                          ERROR   .2              ;MMU ERROR
8383                                                                  ;
8384  036764  022327  125012          14$:    CMP     (R3)+,#125012           ;
8385  036770  001401                          BEQ     15$                     ;
8386  036772  104002                          ERROR   .2              ;MMU ERROR
8387                                                                  ;
8388  036774  020327  172340          15$:    CMP     R3,#172340              ;
8389  037000  001365                          BNE     13$                     ;
8390  037002  022327  125252          16$:    CMP     (R3)+,#125252           ;CHECK KPARS
8391  037006  001401                          BEQ     17$                     ;
8392  037010  104002                          ERROR   .2              ;MMU ERROR
8393                                                                  ;
8394  037012  022327  052525          17$:    CMP     (R3)+,#52525            ;
8395  037016  001401                          BEQ     18$                     ;
8396  037020  104002                          ERROR   .2              ;MMU ERROR
8397                                                                  ;
8398  037022  020327  172400          18$:    CMP     R3,#172400              ;
8399  037026  001365                          BNE     16$                     ;
8400  037030  012703  177600                  MOV     #177600,R3              ;CHECK UPDRS
8401  037034  022327  052404          19$:    CMP     (R3)+,#52404            ;
8402  037040  001401                          BEQ     20$                     ;
8403  037042  104002                          ERROR   .2              ;MMU ERROR
8404                                                                  ;
8405  037044  022327  125012          20$:    CMP     (R3)+,#125012           ;
8406  037050  001401                          BEQ     21$                     ;
8407  037052  104002                          ERROR   .2              ;MMU ERROR
8408                                                                  ;
8409  037054  020327  177640          21$:    CMP     R3,#177640              ;
8410  037060  001365                          BNE     19$                     ;
8411  037062  022327  125252          22$:    CMP     (R3)+,#125252           ;CHECK UPARS
8412  037066  001401                          BEQ     23$                     ;
8413  037070  104002                          ERROR   .2              ;MMU ERROR
8414                                                                  ;
8415  037072  022327  052525          23$:    CMP     (R3)+,#52525            ;
```

```
 8416  037076  001401                    BEQ     24$                     ;
 8417  037100  104002                    ERROR   .2                     ;MMU ERROR
 8418                                                                    ;
 8419  037102  020327  177700    24$:    CMP     R3,#177700              ;
 8420  037106  001365                    BNE     22$                     ;
 8421                              ;
 8422                              ;REVERSE ALTERNATING PATTERN
 8423                              ;
 8424  037110  012700  172200            MOV     #172200,R0              ;LOAD SPDRS WITH REVERSE PATTERN
 8425  037114  012720  125012    25$:    MOV     #125012,(R0)+           ;
 8426  037120  012720  052404            MOV     #52404,(R0)+            ;
 8427  037124  020027  172240            CMP     R0,#172240              ;
 8428  037130  001371                    BNE     25$                     ;
 8429  037132  012720  052525    26$:    MOV     #52525,(R0)+            ;LOAD SPARS WITH REVERSE PATTERN
 8430  037136  012720  125252            MOV     #125252,(R0)+           ;
 8431  037142  020027  172300            CMP     R0,#172300              ;
 8432  037146  001371                    BNE     26$                     ;
 8433  037150  012720  125012    27$:    MOV     #125012,(R0)+           ;LOAD KPDRS WITH REVERSE PATTERN
 8434  037154  012720  052404            MOV     #52404,(R0)+            ;
 8435  037160  020027  172340            CMP     R0,#172340              ;
 8436  037164  001371                    BNE     27$                     ;
 8437  037166  012720  052525    28$:    MOV     #52525,(R0)+            ;LOAD KPARS WITH REVERSE PATTERN
 8438  037172  012720  125252            MOV     #125252,(R0)+           ;
 8439  037176  020027  172400            CMP     R0,#172400              ;
 8440  037202  001371                    BNE     28$                     ;
 8441  037204  012700  177600            MOV     #177600,R0              ;LOAD UPDRS WITH REVERSE PATTERN
 8442  037210  012720  125012    29$:    MOV     #125012,(R0)+           ;
 8443  037214  012720  052404            MOV     #52404,(R0)+            ;
 8444  037220  020027  177640            CMP     R0,#177640              ;
 8445  037224  001371                    BNE     29$                     ;
 8446  037226  012720  052525    30$:    MOV     #52525,(R0)+            ;LOAD UPARS WITH REVERSE PATTERN
 8447  037232  012720  125252            MOV     #125252,(R0)+           ;
 8448  037236  020027  177700            CMP     R0,#177700              ;
 8449  037242  001371                    BNE     30$                     ;
 8450                              ;
 8451  037244  012703  172200            MOV     #172200,R3              ;CHECK SPDRS
 8452  037250  022327  125012    31$:    CMP     (R3)+,#125012           ;
 8453  037254  001401                    BEQ     32$                     ;
 8454  037256  104002                    ERROR   .2                      ;MMU ERROR
 8455                                                                    ;
 8456  037260  022327  052404    32$:    CMP     (R3)+,#52404            .
 8457  037264  001401                    BEQ     33$                     ;
 8458  037266  104002                    ERROR   .2                      ;MMU ERROR
 8459                                                                    ;
 8460  037270  020327  172240    33$:    CMP     R3,#172240              ;
 8461  037274  001365                    BNE     31$                     ;
 8462  037276  022327  052525    34$:    CMP     (R3)+,#52525            ;CHECK SPARS
 8463  037302  001401                    BEQ     35$                     ;
 8464  037304  104002                    ERROR   .2                      ;MMU ERROR
 8465                                                                    ;
 8466  037306  022327  125252    35$:    CMP     (R3)+,#125252           ;
 8467  037312  001401                    BEQ     36$                     ;
 8468  037314  104002                    ERROR   .2                      ;MMU ERROR
 8469                                                                    ;
 8470  037316  020327  172300    36$:    CMP     R3,#172300              ;
 8471  037322  001365                    BNE     34$                     ;
```

```
 8472  037324  022327  125012          37$:    CMP     (R3)+,#125012           ;CHECK KPDRS
 8473  037330  001401                          BEQ     38$                     ;
 8474  037332  104002                          ERROR   .+2             ;MMU ERROR
 8475                                                                  ;
 8476  037334  022327  052404          38$:    CMP     (R3)+,#52404            ;
 8477  037340  001401                          BEQ     39$                     ;
 8478  037342  104002                          ERROR   .+2             ;MMU ERROR
 8479                                                                  ;
 8480  037344  020327  172340          39$:    CMP     R3,#172340              ;
 8481  037350  001365                          BNE     37$                     ;
 8482  037352  022327  052525          40$:    CMP     (R3)+,#52525            ;CHECK KPARS
 8483  037356  001401                          BEQ     41$                     ;
 8484  037360  104002                          ERROR   .+2             ;MMU ERROR
 8485                                                                  ;
 8486  037362  022327  125252          41$:    CMP     (R3)+,#125252           ;
 8487  037366  001401                          BEQ     42$                     ;
 8488  037370  104002                          ERROR   .+2             ;MMU ERROR
 8489                                                                  ;
 8490  037372  020327  172400          42$:    CMP     R3,#172400              ;
 8491  037376  001365                          BNE     40$                     ;
 8492  037400  012703  177600                  MOV     #177600,R3              ;CHECK UPDRS
 8493  037404  022327  125012          43$:    CMP     (R3)+,#125012           ;
 8494  037410  001401                          BEQ     44$                     ;
 8495  037412  104002                          ERROR   .+2             ;MMU ERROR
 8496                                                                  ;
 8497  037414  022327  052404          44$:    CMP     (R3)+,#52404            ;
 8498  037420  001401                          BEQ     45$                     ;
 8499  037422  104002                          ERROR   .+2             ;MMU ERROR
 8500                                                                  ;
 8501  037424  020327  177640          45$:    CMP     R3,#177640              ;
 8502  037430  001365                          BNE     43$                     ;
 8503  037432  022327  052525          46$:    CMP     (R3)+,#52525            ;CHECK UPARS
 8504  037436  001401                          BEQ     47$                     ;
 8505  037440  104002                          ERROR   .+2             ;MMU ERROR
 8506                                                                  ;
 8507  037442  022327  125252          47$:    CMP     (R3)+,#125252           ;
 8508  037446  001401                          BEQ     48$                     ;
 8509  037450  104002                          ERROR   .+2             ;MMU ERROR
 8510                                                                  ;
 8511  037452  020327  177700          48$:    CMP     R3,#177700              ;
 8512  037456  001365                          BNE     46$                     ;
 8513
 8514  037460                          TSMMU5:
 8515                                  ;       TEST MMR0 ABORT BITS
 8516  037460  012737  160000  177572            MOV     #160000,@#177572        ;LOAD MMR0<15:13>=111
 8517  037466  005067  143304                    CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 8518  037472  013700  177572                    MOV     @#SR0,R0                ;SAVE SR0 IN R0
 8519  037476  042700  000176                    BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0
 8520  037502  020027  160000                    CMP     R0,#160000              ;CHECK MMR0
 8521  037506  001401                          BEQ     1$                      ;
 8522  037510  104002                          ERROR   .+2             ;MMU ERROR
 8523                                                                  ;
 8524  037512  005037  177572          1$:     CLR     @#177572                ;LOAD MMR0=0
 8525  037516  013700  177572                    MOV     @#SR0,R0                ;SAVE SR0 IN R0
 8526  037522  042700  000176                    BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0
 8527  037526  020027  000000                    CMP     R0,#0                   ;CHECK MMR0
```

```
 8528  037532  001401                          BEQ     2$                      ;
 8529  037534  104002                          ERROR   .+2                 ;MMU ERROR
 8530                                                                        ;
 8531  037536  012737  120000  177572  2$:     MOV     #120000,@#177572        ;LOAD MMR0<15:13>=101
 8532  037544  013700  177572                  MOV     @#SR0,R0                ;SAVE SR0 IN R0
 8533  037550  042700  000176                  BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0.
 8534  037554  020027  120000                  CMP     R0,#120000              ;CHECK MMR0
 8535  037560  001401                          BEQ     3$                      ;
 8536  037562  104002                          ERROR   .+2                 ;MMU ERROR
 8537                                                                        ;
 8538  037564  012737  040000  177572  3$:     MOV     #40000,@#177572         ;LOAD MMR0<15:13>=010
 8539  037572  013700  177572                  MOV     @#SR0,R0                ;SAVE SR0 IN R0
 8540  037576  042700  000176                  BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0.
 8541  037602  020027  040000                  CMP     R0,#40000               ;CHECK MMR0
 8542  037606  001401                          BEQ     4$                      ;
 8543  037610  104002                          ERROR   .+2                 ;MMU ERROR
 8544  037612                          4$:
 8545  037612                          TSMMU6:
 8546                                  ;       TEST MMR3 BITS 5-0
 8547  037612  005037  177572                  CLR     @#177572                ;MMU OFF
 8548  037616  005067  143154                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 8549  037622  012737  000077  172516          MOV     #77,@#172516            ;LOAD MMR3<5:0>=77
 8550  037630  023727  172516  000077          CMP     @#172516,#77            ;CHECK MMR3
 8551  037636  001401                          BEQ     1$                      ;
 8552  037640  104002                          ERROR   .+2                 ;MMU ERROR
 8553  037642  005037  172516          1$:     CLR     @#172516                ;LOAD MMR3<5:0>=0
 8554  037646  023727  172516  000000          CMP     @#172516,#0             ;CHECK MMR3
 8555  037654  001401                          BEQ     2$                      ;
 8556  037656  104002                          ERROR   .+2                 ;MMU ERROR
 8557  037660  012737  000052  172516  2$:     MOV     #52,@#172516            ;LOAD MMR3<5:0>=52
 8558  037666  023727  172516  000052          CMP     @#172516,#52            ;CHECK MMR3
 8559  037674  001401                          BEQ     3$                      ;
 8560  037676  104002                          ERROR   .+2                 ;MMU ERROR
 8561  037700  012737  000025  172516  3$:     MOV     #25,@#172516            ;LOAD MMR3<5:0>=25
 8562  037706  023727  172516  000025          CMP     @#172516,#25            ;CHECK MMR3
 8563  037714  001401                          BEQ     4$                      ;
 8564  037716  104002                          ERROR   .+2                 ;MMU ERROR
 8565  037720                          4$:
 8566  037720                          TSMM6A:
 8567                                  ;       TEST MFPI (MOVE FROM PREVIOUS INST SPACE)
 8568  037720  005037  177572                  CLR     @#177572                ;MMU OFF
 8569  037724  005037  002776                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
 8570  037730  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
 8571  037736  012706  001000                  MOV     #STBOT,SP               ;INIT THE USER STACK POINTER
 8572  037742  010637  003002                  MOV     R6,@#SAVUSE             ;SAVE USER SP
 8573  037746  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 8574  037754  012706  001000                  MOV     #STBOT,SP               ;INIT THE SUPERVISOR STACK POINTER
 8575  037760  010637  003000                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
 8576  037764  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 8577  037772  004767  072346                  JSR     PC,MMU                  ;INIT MMU
 8578  037776  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
 8579  040004  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
 8580  040010  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON TOP OF STACK
 8581  040014  012737  135072  000244          MOV     #135072,@#244           ;SETUP DATA AT TEST LOCATION
 8582  040022  012767  077400  137570          MOV     #77400,UDPDR0           ;SETUP UDPDR0 TO ABORT
 8583  040030  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
```

```
8584  040034  005237  177572          INC     @#177572                ;TURN MMU ON
8585  040040  006523                  MFPI    (R3)+                   ; TEST INSTRUCTION
8586  040042  022737  030010  177776  CMP     #030010,@#177776        ;IS PSW CORRECT
8587  040050  001401                  BEQ     1$                      ;YES GO ON
8588  040052  104002                  ERROR   .2                  ;MMU ERROR
8589                                                              ;NO GO TO ERROR
8590  040054  005037  177572  1$:     CLR     @#177572                ;TURN MMU OFF
8591  040060  012737  140000  177776  MOV     #140000,@#177776        ;POINT TO USER SPACE
8592  040066  020637  003002          CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8593  040072  001401                  BEQ     100$                    ;YES GO ON
8594  040074  104002                  ERROR   .2                  ;MMU ERROR
8595                                                              ;NO GO TO ERROR
8596  040076  012737  040000  177776  100$:   MOV     #040000,@#177776        ;POINT TO SUPERVISOR SPACE
8597  040104  020637  003000          CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8598  040110  001401                  BEQ     200$                    ;YES GO ON
8599  040112  104002                  ERROR   .2                  ;MMU ERROR
8600                                                              ;NO GO TO ERROR
8601  040114  023727  000244  135072  200$:   CMP     @#244,#135072           ;IS TEST DATA OK
8602  040122  001401                  BEQ     2$                      ;YES GO ON
8603  040124  104002                  ERROR   .2                  ;MMU ERROR
8604                                                              ;NO GO TO ERROR
8605  040126  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
8606  040132  001401                  BEQ     3$                      ;YES GO ON
8607  040134  104002                  ERROR   .2                  ;MMU ERROR
8608                                                              ;NO GO TO ERROR
8609  040136  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
8610  040142  022627  135072          CMP     (SP)+,#135072           ;IS KERNEL STACK CORRECT
8611  040146  001401                  BEQ     4$                      ;YES GO ON
8612  040150  104002                  ERROR   .2                  ;MMU ERROR
8613                                                              ;NO GO TO ERROR
8614  040152  021627  177777          4$:     CMP     (SP),#177777            ;IS STACK CORRECT
8615  040156  001401                  BEQ     5$                      ;YES GO ON
8616  040160  104002                  ERROR   .2                  ;MMU ERROR
8617                                                              ;NO GO TO ERROR
8618  040162  012737  030017  177776  5$:     MOV     #030017,@#177776        ;SETUP PSW
8619  040170  012737  173621  000244          MOV     #173621,@#244           ;SETUP TEST LOCATION
8620  040176  012701  000244                  MOV     #244,R1                 ;SETUP R1
8621  040202  005237  177572                  INC     @#177572                ;TURN MMU ON
8622  040206  006511                  MFPI    (R1)                    ;TEST INSTRUCTION
8623  040210  022737  030011  177776  CMP     #030011,@#177776        ;IS PSW CORRECT
8624  040216  001401                  BEQ     300$                    ;YES GO ON
8625  040220  104002                  ERROR   .2                  ;MMU ERROR
8626                                                              ;NO GO TO ERROR
8627  040222  005037  177572  300$:   CLR     @#177572                ;TURN MMU OFF
8628  040226  023727  000244  173621  CMP     @#244,#173621           ;IS TEST LOCATION CORRECT
8629  040234  001401                  BEQ     301$                    ;YES GO ON
8630  040236  104002                  ERROR   .2                  ;MMU ERROR
8631                                                              ;NO GO TO ERROR
8632  040240  020127  000244  301$:   CMP     R1,#244                 ;IS R1 CORRECT
8633  040244  001401                  BEQ     302$                    ;YES GO ON
8634  040246  104002                  ERROR   .2                  ;MMU ERROR
8635                                                              ;NO GO TO ERROR
8636  040250  005037  177776  302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8637  040254  022627  173621          CMP     (SP)+,#173621           ;IS STACK CORRECT
8638  040260  001401                  BEQ     303$                    ;YES GO ON
8639  040262  104002                  ERROR   .2                  ;MMU ERROR
```

```
8640
8641  040264  021627  177777          3038:   CMP     (SP),#177777            ;NO GO TO ERROR
8642  040270  001401                          BEQ     3048                    ;IS STACK CORRECT
8643  040272  104002                          ERROR   .2                      ;YES GO ON
8644                                                                          ;MMU ERROR
8645  040274  005003                  3048:   CLR     R3                      ;NO GO TO ERROR
8646  040276  005237  177572                  INC     @#177572                ;SETUP SOURCE FOR NEXT TEST
8647  040302  006503                          MFPI    R3                      ;TURN MMU ON
8648  040304  022737  000004  177776          CMP     #4,@#177776             ; TEST INSTRUCTION
8649  040312  001401                          BEQ     68                      ;IS PSW CORRECT
8650  040314  104002                          ERROR   .2                      ;YES GO ON
8651                                                                          ;MMU ERROR
8652  040316  005037  177572          68:     CLR     @#177572                ;NO GO TO ERROR
8653  040322  020327  000000                  CMP     R3,#0                   ;TURN MMU OFF
8654  040326  001401                          BEQ     78                      ;IS R3 CORRECT
8655  040330  104002                          ERROR   .2                      ;YES GO ON
8656                                                                          ;MMU ERROR
8657  040332  022627  000000          78:     CMP     (SP)+,#0                ;NO GO TO ERROR
8658  040336  001401                          BEQ     88                      ;IS STACK CORRECT
8659  040340  104002                          ERROR   .2                      ;YES GO ON
8660                                                                          ;MMU ERROR
8661  040342  022627  177777          88:     CMP     (SP)+,#177777           ;NO GO TO ERROR
8662  040346  001401                          BEQ     98                      ;IS STACK CORRECT
8663  040350  104002                          ERROR   .2                      ;YES GO ON
8664                                                                          ;MMU ERROR
8665  040352  012637  000244          98:     MOV     (SP)+,@#244             ;NO GO TO ERROR
8666                                                                          ;RESTORE TEST LOCATION
8667                                  ;
8668  040356                          TSMM68:
8669                                  ;       TEST MFPD (MOVE FROM PREVIOUS DATA SPACE)
8670  040356  005037  177572                  CLR     @#177572                ;MMU OFF
8671  040362  005037  002776                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
8672  040366  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8673  040374  010637  003002                  MOV     R6,@#SAVUSE             ;SAVE USER SP
8674  040400  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8675  040406  010637  003000                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
8676  040412  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
8677  040420  004767  071720                  JSR     PC,MMU                  ;INIT MMU
8678  040424  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
8679  040432  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
8680  040436  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON TOP OF STACK
8681  040442  012737  157002  000244          MOV     #157002,@#244           ;SETUP DATA AT TEST LOCATION
8682  040450  012767  077400  137122          MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
8683  040456  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
8684  040462  005237  177572                  INC     @#177572                ;TURN MMU ON
8685  040466  106523                          MFPD    (R3)+                   ; TEST INSTRUCTION
8686  040470  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
8687  040476  001401                          BEQ     18                      ;YES GO ON
8688  040500  104002                          ERROR   .2                      ;MMU ERROR
8689                                                                          ;NO GO TO ERROR
8690  040502  005037  177572          18:     CLR     @#177572                ;TURN MMU OFF
8691  040506  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8692  040514  020637  003002                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8693  040520  001401                          BEQ     1008                    ;YES GO ON
8694  040522  104002                          ERROR   .2                      ;MMU ERROR
8695                                                                          ;NO GO TO ERROR
```

```
8696  040524  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8697  040532  020637  003000                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8698  040536  001401                          BEQ     200$                    ;YES GO ON
8699  040540  104002                          ERROR   .2                      ;MMU ERROR
8700                                                                          ;NO GO TO ERROR
8701  040542  023727  000244  157002  200$:   CMP     @#244,#157002           ;IS TEST DATA OK
8702  040550  001401                          BEQ     2$                      ;YES GO ON
8703  040552  104002                          ERROR   .2                      ;MMU ERROR
8704                                                                          ;NO GO TO ERROR
8705  040554  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
8706  040560  001401                          BEQ     3$                      ;YES GO ON
8707  040562  104002                          ERROR   .2                      ;MMU ERROR
8708                                                                          ;NO GO TO ERROR
8709  040564  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
8710  040570  022627  157002                  CMP     (SP)+,#157002           ;IS KERNEL STACK CORRECT
8711  040574  001401                          BEQ     4$                      ;YES GO ON
8712  040576  104002                          ERROR   .2                      ;MMU ERROR
8713                                                                          ;NO GO TO ERROR
8714  040600  021627  177777          4$:     CMP     (SP),#177777            ;IS STACK CORRECT
8715  040604  001401                          BEQ     5$                      ;YES GO ON
8716  040606  104002                          ERROR   .2                      ;MMU ERROR
8717                                                                          ;NO GO TO ERROR
8718  040610  012737  030017  177776  5$:     MOV     #30017,@#177776         ;SETUP PSW
8719  040616  012737  103456  000244          MOV     #103456,@#244           ;SETUP TEST LOCATION
8720  040624  012701  000244                  MOV     #244,R1                 ;SETUP R1
8721  040630  005237  177572                  INC     @#177572                ;TURN MMU ON
8722  040634  106511                          MFPD    (R1)                    ;TEST INSTRUCTION
8723  040636  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8724  040644  001401                          BEQ     300$                    ;YES GO ON
8725  040646  104002                          ERROR   .2                      ;MMU ERROR
8726                                                                          ;NO GO TO ERROR
8727  040650  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
8728  040654  023727  000244  103456          CMP     @#244,#103456           ;IS TEST LOCATION CORRECT
8729  040662  001401                          BEQ     301$                    ;YES GO ON
8730  040664  104002                          ERROR   .2                      ;MMU ERROR
8731                                                                          ;NO GO TO ERROR
8732  040666  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8733  040672  001401                          BEQ     302$                    ;YES GO ON
8734  040674  104002                          ERROR   .2                      ;MMU ERROR
8735                                                                          ;NO GO TO ERROR
8736  040676  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8737  040702  022627  103456                  CMP     (SP)+,#103456           ;IS STACK CORRECT
8738  040706  001401                          BEQ     303$                    ;YES GO ON
8739  040710  104002                          ERROR   .2                      ;MMU ERROR
8740                                                                          ;NO GO TO ERROR
8741  040712  021627  177777          303$:   CMP     (SP),#177777            ;IS STACK CORRECT
8742  040716  001401                          BEQ     304$                    ;YES GO ON
8743  040720  104002                          ERROR   .2                      ;MMU ERROR
8744                                                                          ;NO GO TO ERROR
8745  040722  012737  030017  177776  304$:   MOV     #30017,@#177776         ;SETUP PSW
8746  040730  012737  113672  000244          MOV     #1136:2,@#244           ;SETUP TEST LOCATION
8747  040736  012701  000246                  MOV     #246,R1                 ;SETUP R1
8748  040742  005237  177572                  INC     @#177572                ;TURN MMU ON
8749  040746  106541                          MFPD    -(R1)                   ;TEST INSTRUCTION
8750  040750  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8751  040756  001401                          BEQ     400$                    ;YES GO ON
```

```
8752  040760  104002                          ERROR   ;2              ;MMU ERROR
8753                                                                  ;NO GO TO ERROR
8754  040762  005037  177572          400$:   CLR     @#177572        ;TURN MMU OFF
8755  040766  023727  000244  113672          CMP     @#244,#113672   ;IS TEST LOCATION CORRECT
8756  040774  001401                          BEQ     401$            ;YES GO ON
8757  040776  104002                          ERROR   ;2              ;MMU ERROR
8758                                                                  ;NO GO TO ERROR
8759  041000  020127  000244          401$:   CMP     R1,#244         ;IS R1 CORRECT
8760  041004  001401                          BEQ     402$            ;YES GO ON
8761  041006  104002                          ERROR   ;2              ;MMU ERROR
8762                                                                  ;NO GO TO ERROR
8763  041010  005037  177776          402$:   CLR     @#177776        ;SET PSW TO KERNEL MODE
8764  041014  022627  113672                  CMP     (SP)+,#113672   ;IS STACK CORRECT
8765  041020  001401                          BEQ     403$            ;YES GO ON
8766  041022  104002                          ERROR   ;2              ;MMU ERROR
8767                                                                  ;NO GO TO ERROR
8768  041024  021627  177777          403$:   CMP     (SP),#177777    ;IS STACK CORRECT
8769  041030  001401                          BEQ     404$            ;YES GO ON
8770  041032  104002                          ERROR   ;2              ;MMU ERROR
8771                                                                  ;NO GO TO ERROR
8772  041034  005003                  404$:   CLR     R3              ;SETUP SOURCE FOR NEXT TEST
8773  041036  005237  177572                  INC     @#177572        ;TURN MMU ON
8774  041042  106503                          MFPD    R3              ; TEST INSTRUCTION
8775  041044  022737  000004  177776          CMP     #4,@#177776     ;IS PSW CORRECT
8776  041052  001401                          BEQ     6$              ;YES GO ON
8777  041054  104002                          ERROR   ;2              ;MMU ERROR
8778                                                                  ;NO GO TO ERROR
8779  041056  005037  177572          6$:     CLR     @#177572        ;TURN MMU OFF
8780  041062  020327  000000                  CMP     R3,#0           ;IS R3 CORRECT
8781  041066  001401                          BEQ     7$              ;YES GO ON
8782  041070  104002                          ERROR   ;2              ;MMU ERROR
8783                                                                  ;NO GO TO ERROR
8784  041072  022627  000000          7$:     CMP     (SP)+,#0        ;IS STACK CORRECT
8785  041076  001401                          BEQ     8$              ;YES GO ON
8786  041100  104002                          ERROR   ;2              ;MMU ERROR
8787                                                                  ;NO GO TO ERROR
8788  041102  022627  177777          8$:     CMP     (SP)+,#177777   ;IS STACK CORRECT
8789  041106  001401                          BEQ     9$              ;YES GC ON
8790  041110  104002                          ERROR   ;2              ;MMU ERROR
8791                                                                  ;NO GO TO ERROR
8792  041112  012637  000244          9$:     MOV     (SP)+,@#244     ;RESTORE TEST LOCATION
8793
8794                                  ;
8795  041116                          TSMM6C:
8796                                  ;       TEST MTPI (MOVE TO PREVIOUS INSTRUCTION SPACE)
8797  041116  005037  177572                  CLR     @#177572        ;MMU OFF
8798  041122  005037  002776                  CLR     @#FLAG          ;CLEAR MMU ABORT FLAG
8799  041126  012737  140000  177776          MOV     #140000,@#177776 ;POINT TO USER SPACE
8800  041134  010637  003002                  MOV     R6,@#SAVUSE     ;SAVE USER SP
8801  041140  012737  040000  177776          MOV     #40000,@#177776 ;POINT TO SUPERVISOR SPACE
8802  041146  010637  003000                  MOV     R6,@#SAVSUP     ;SAVE SUPERVISOR SP
8803  041152  012737  030000  177776          MOV     #30000,@#177776 ;SETUP PSW
8804  041160  004767  071160                  JSR     PC,MMU          ;INIT MMU
8805  041164  012737  000027  172516          MOV     #27,@#172516    ;SETUP MMR3
8806  041172  013746  000244                  MOV     @#244,-(SP)     ;SAVE DATA AT TEST LOCATION
8807  041176  012746  177777                  MOV     #177777,-(SP)   ;PUT KNOWN DATA ON STACK
```

```
8808  041202  012746  120413                  MOV     #120413,-(SP)           ;PUT TEST DATA ON STACK
8809  041206  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
8810  041214  012767  077400  136376          MOV     #77400,UDPDRO           ;SETUP UDPDRO TO ABORT
8811  041222  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
8812  041226  005237  177572                  INC     @#177572                ;TURN MMU ON
8813  041232  006623                          MTPI    (R3)+                   ; TEST INSTRUCTION
8814  041234  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
8815  041242  001401                          BEQ     1$                      ;YES GO ON
8816  041244  104002                          ERROR   ^2              ;MMU ERROR
8817                                                                  ;NO GO TO ERROR
8818  041246  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
8819  041252  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8820  041260  020637  003002                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8821  041264  001401                          BEQ     100$                    ;YES GO ON
8822  041266  104002                          ERROR   ^2              ;MMU ERROR
8823                                                                  ;NO GO TO ERROR
8824  041270  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8825  041276  020637  003000                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8826  041302  001401                          BEQ     200$                    ;YES GO ON
8827  041304  104002                          ERROR   ^2              ;MMU ERROR
8828                                                                  ;NO GO TO ERROR
8829  041306  023727  000244  120413  200$:   CMP     @#244,#120413           ;IS TEST LOCATION CORRECT
8830  041314  001401                          BEQ     2$                      ;YES GO ON
8831  041316  104002                          ERROR   ^2              ;MMU ERROR
8832                                                                  ;NO GO TO ERROR
8833  041320  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
8834  041324  001401                          BEQ     3$                      ;YES GO ON
8835  041326  104002                          ERROR   ^2              ;MMU ERROR
8836                                                                  ;NO GO TO ERROR
8837  041330  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
8838  041334  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
8839  041340  001401                          BEQ     4$                      ;YES GO ON
8840  041342  104002                          ERROR   ^2              ;MMU ERROR
8841                                                                  ;NO GO TO ERROR
8842  041344  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
8843  041352  012746  145121                  MOV     #145121,-(SP)           ;SETUP TEST DATA
8844  041356  012701  000244                  MOV     #244,R1                 ;SETUP R1
8845  041362  005237  177572                  INC     @#177572                ;TURN MMU ON
8846  041366  006611                          MTPI    (R1)                    ;TEST INSTRUCTION
8847  041370  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8848  041376  001401                          BEQ     300$                    ;YES GO ON
8849  041400  104002                          ERROR   ^2              ;MMU ERROR
8850                                                                  ;NO GO TO ERROR
8851  041402  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
8852  041406  023727  000244  145121          CMP     @#244,#145121           ;IS TEST LOCATION CORRECT
8853  041414  001401                          BEQ     301$                    ;YES GO ON
8854  041416  104002                          ERROR   ^2              ;MMU ERROR
8855                                                                  ;NO GO TO ERROR
8856  041420  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8857  041424  001401                          BEQ     302$                    ;YES GO ON
8858  041426  104002                          ERROR   ^2              ;MMU ERROR
8859                                                                  ;NO GO TO ERROR
8860  041430  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8861  041434  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
8862  041440  001401                          BEQ     304$                    ;YES GO ON
8863  041442  104002                          ERROR   ^2              ;MMU ERROR
```

```
8864
8865  041444  012737  030017  177776  304$:   MOV     #30017,@#177776     ;NO GO TO ERROR
8866  041452  012746  122347                  MOV     #122347,-(SP)       ;SETUP PSW
8867  041456  012701  000246                  MOV     #246,R1             ;SETUP TEST DATA
8868  041462  005237  177572                  INC     @#177572            ;SETUP R1
8869  041466  006641                          MTPI    -(R1)               ;TURN MMU ON
8870  041470  022737  030011  177776          CMP     #30011,@#177776     ;TEST INSTRUCTION
8871  041476  001401                          BEQ     400$                ;IS PSW CORRECT
8872  041500  104002                          ERROR   .+2                 ;YES GO ON
8873                                                                      ;MMU ERROR
8874  041502  005037  177572          400$:   CLR     @#177572            ;NO GO TO ERROR
8875  041506  023727  000244  122347          CMP     @#244,#122347       ;TURN MMU OFF
8876  041514  001401                          BEQ     401$                ;IS TEST LOCATION CORRECT
8877  041516  104002                          ERROR   .+2                 ;YES GO ON
8878                                                                      ;MMU ERROR
8879  041520  020127  000244          401$:   CMP     R1,#244             ;NO GO TO ERROR
8880  041524  001401                          BEQ     402$                ;IS R1 CORRECT
8881  041526  104002                          ERROR   .+2                 ;YES GO ON
8882                                                                      ;MMU ERROR
8883  041530  005037  177776          402$:   CLR     @#177776            ;NO GO TO ERROR
8884  041534  021627  177777                  CMP     (SP),#177777        ;SET PSW TO KERNEL MODE
8885  041540  001401                          BEQ     404$                ;IS STACK CORRECT
8886  041542  104002                          ERROR   .+2                 ;YES GO ON
8887                                                                      ;MMU ERROR
8888  041544  005046                  404$:   CLR     -(SP)               ;NO GO TO ERROR
8889  041546  005237  177572                  INC     @#177572            ;SETUP STACK FOR NEXT TEST
8890  041552  006603                          MTPI    R3                  ;TURN MMU ON
8891  041554  022737  000004  177776          CMP     #4,@#177776         ; TEST INSTRUCTION
8892  041562  001401                          BEQ     5$                  ;IS PSW CORRECT
8893  041564  104002                          ERROR   .+2                 ;YES GO ON
8894                                                                      ;MMU ERROR
8895  041566  005037  177572            5$:   CLR     @#177572            ;NO GO TO ERROR
8896  041572  020327  000000                  CMP     R3,#0               ;TURN MMU OFF
8897  041576  001401                          BEQ     6$                  ;IS R3 CORRECT
8898  041600  104002                          ERROR   .+2                 ;YES GO ON
8899                                                                      ;MMU ERROR
8900  041602  022627  177777            6$:   CMP     (SP)+,#177777       ;NO GO TO ERROR
8901  041606  001401                          BEQ     7$                  ;IS STACK CORRECT
8902  041610  104002                          ERROR   .+2                 ;YES GO ON
8903                                                                      ;MMU ERROR
8904  041612  012637  000244            7$:   MOV     (SP)+,@#244         ;NO GO TO ERROR
8905                                                                      ;RESTORE TEST LOCATION
8906                                  ;
8907  041616                          TSMM6D:
8908                                  ;       TEST MTPD (MOVE TO PREVIOUS DATA SPACE)
8909  041616  005037  177572                  CLR     @#177572            ;MMU OFF
8910  041622  005037  002776                  CLR     @#FLAG              ;CLEAR MMU ABORT FLAG
8911  041626  012737  140000  177776          MOV     #140000,@#177776    ;POINT TO USER SPACE
8912  041634  010637  003002                  MOV     R6,@#SAVUSE         ;SAVE USER SP
8913  041640  012737  040000  177776          MOV     #40000,@#177776     ;POINT TO SUPERVISOR SPACE
8914  041646  010637  003000                  MOV     R6,@#SAVSUP         ;SAVE SUPERVISOR SP
8915  041652  012737  030000  177776          MOV     #30000,@#177776     ;SETUP PSW
8916  041660  004767  070460                  JSR     PC,MMU              ;INIT MMU
8917  041664  012737  000027  172516          MOV     #27,@#172516        ;SETUP MMR3
8918  041672  013746  000244                  MOV     @#244,-(SP)         ;SAVE DATA AT TEST LOCATION
8919  041676  012746  177777                  MOV     #177777,-(SP)       ;PUT KNOWN DATA ON STACK
```

```
8920  041702  012746  100004                  MOV     #100004,-(SP)           ;PUT TEST DATA ON STACK
8921  041706  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
8922  041714  012767  077400  135656          MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
8923  041722  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
8924  041726  005237  177572                  INC     @#177572                ;TURN MMU ON
8925  041732  106623                          MTPD    (R3)+                   ; TEST INSTRUCTION
8926  041734  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
8927  041742  001401                          BEQ     1$                      ;YES GO ON
8928  041744  104002                          ERROR   .+2             ;MMU ERROR
8929                                                                  ;NO GO TO ERROR
8930  041746  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
8931  041752  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8932  041760  020637  003002                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8933  041764  001401                          BEQ     100$                    ;YES GO ON
8934  041766  104002                          ERROR   .+2             ;MMU ERROR
8935                                                                  ;NO GO TO ERROR
8936  041770  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8937  041776  020637  003000                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8938  042002  001401                          BEQ     200$                    ;YES GO ON
8939  042004  104002                          ERROR   .+2             ;MMU ERROR
8940                                                                  ;NO GO TO ERROR
8941  042006  023727  000244  100004  200$:   CMP     @#244,#100004           ;IS TEST LOCATION CORRECT
8942  042014  001401                          BEQ     2$                      ;YES GO ON
8943  042016  104002                          ERROR   .+2             ;MMU ERROR
8944                                                                  ;NO GO TO ERROR
8945  042020  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
8946  042024  001401                          BEQ     3$                      ;YES GO ON
8947  042026  104002                          ERROR   .+2             ;MMU ERROR
8948                                                                  ;NO GO TO ERROR
8949  042030  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
8950  042034  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
8951  042040  001401                          BEQ     4$                      ;YES GO ON
8952  042042  104002                          ERROR   .+2             ;MMU ERROR
8953                                                                  ;NO GO TO ERROR
8954  042044  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
8955  042052  012746  100737                  MOV     #100737,-(SP)           ;SETUP TEST DATA
8956  042056  012701  000244                  MOV     #244,R1                 ;SETUP R1
8957  042062  005237  177572                  INC     @#177572                ;TURN MMU ON
8958  042066  106611                          MTPD    (R1)                    ;TEST INSTRUCTION
8959  042070  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8960  042076  001401                          BEQ     300$                    ;YES GO ON
8961  042100  104002                          ERROR   .+2             ;MMU ERROR
8962                                                                  ;NO GO TO ERROR
8963  042102  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
8964  042106  023727  000244  100737          CMP     @#244,#100737           ;IS TEST LOCATION CORRECT
8965  042114  001401                          BEQ     301$                    ;YES GO ON
8966  042116  104002                          ERROR   .+2             ;MMU ERROR
8967                                                                  ;NO GO TO ERROR
8968  042120  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8969  042124  001401                          BEQ     302$                    ;YES GO ON
8970  042126  104002                          ERROR   .+2             ;MMU ERROR
8971                                                                  ;NO GO TO ERROR
8972  042130  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8973  042134  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
8974  042140  001401                          BEQ     304$                    ;YES GO ON
8975  042142  104002                          ERROR   .+2             ;MMU ERROR
```

```
8976
8977  042144  012737  030017  177776  304$:   MOV     #30017,@#177776         ;NO GO TO ERROR
8978  042152  012746  156711                  MOV     #156711,-(SP)           ;SETUP PSW
8979  042156  012701  000246                  MOV     #246,R1                 ;SETUP TEST DATA
8980  042162  005237  177572                  INC     @#177572                ;SETUP R1
8981  042166  106641                          MTPD    -(R1)                   ;TURN MMU ON
8982  042170  022737  030011  177776          CMP     #30011,@#177776         ;TEST INSTRUCTION
8983  042176  001401                          BEQ     400$                    ;IS PSW CORRECT
8984  042200  104002                          ERROR   .2                      ;YES GO ON
8985                                                                          ;MMU ERROR
8986  042202  005037  177572          400$:   CLR     @#177572                ;NO GO TO ERROR
8987  042206  023727  000244  156711          CMP     @#244,@#156711          ;TURN MMU OFF
8988  042214  001401                          BEQ     401$                    ;IS TEST LOCATION CORRECT
8989  042216  104002                          ERROR   .2                      ;YES GO ON
8990                                                                          ;MMU ERROR
8991  042220  020127  000244          401$:   CMP     R1,#244                 ;NO GO TO ERROR
8992  042224  001401                          BEQ     402$                    ;IS R1 CORRECT
8993  042226  104002                          ERROR   .2                      ;YES GO ON
8994                                                                          ;MMU ERROR
8995  042230  005037  177776          402$:   CLR     @#177776                ;NO GO TO ERROR
8996  042234  021627  177777                  CMP     (SP),#177777            ;SET PSW TO KERNEL MODE
8997  042240  001401                          BEQ     404$                    ;IS STACK CORRECT
8998  042242  104002                          ERROR   .2                      ;YES GO ON
8999                                                                          ;MMU ERROR
9000  042244  005046                  404$:   CLR     -(SP)                   ;NO GO TO ERROR
9001  042246  005237  177572                  INC     @#177572                ;SETUP STACK FOR NEXT TEST
9002  042252  106603                          MTPD    R3                      ;TURN MMU ON
9003  042254  022737  000004  177776          CMP     #4,@#177776             ; TEST INSTRUCTION
9004  042262  001401                          BEQ     5$                      ;IS PSW CORRECT
9005  042264  104002                          ERROR   .2                      ;YES GO ON
9006                                                                          ;MMU ERROR
9007  042266  005037  177572          5$:     CLR     @#177572                ;NO GO TO ERROR
9008  042272  020327  000000                  CMP     R3,#0                   ;TURN MMU OFF
9009  042276  001401                          BEQ     6$                      ;IS R3 CORRECT
9010  042300  104002                          ERROR   .2                      ;YES GO ON
9011                                                                          ;MMU ERROR
9012  042302  022627  177777          6$:     CMP     (SP)+,#177777           ;NO GO TO ERROR
9013  042306  001401                          BEQ     7$                      ;IS STACK CORRECT
9014  042310  104002                          ERROR   .2                      ;YES GO ON
9015                                                                          ;MMU ERROR
9016  042312  012637  000244          7$:     MOV     (SP)+,@#244             ;NO GO TO ERROR
9017                                                                          ;RESTORE TEST LOCATION
9018                                  ;
9019  042316                          TSMMU7:
9020                                  ;       TEST NON-RESIDENT ABORT
9021  042316  005037  177572                  CLR     @#177572                ;MMU OFF
9022  042322  005067  140450                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
9023  042326  013746  000214                  MOV     @#214,-(SP)             ;SAVE DATA AT TEST LOCATIONS
9024  042332  013746  000216                  MOV     @#216,-(SP)             ;
9025  042336  005067  140442                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9026  042342  005067  140440                  CLR     SAVMR1                  ;
9027  042346  005067  140436                  CLR     SAVMR2                  ;
9028  042352  004767  067766                  JSR     PC,MMU                  ;INIT MMU
9029  042356  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
9030  042364  012702  000200                  MOV     #200,R2                 ;
9031  042370  012737  077400  177600          MOV     #77400,@#177600         ;SETUP FOR AN ABORT
```

J13


```
9032  042376  004767  000164                  JSR     PC,TS7          ;CAUSE AN ABORT TO OCCUR AND
9033                                                                    ;THEN CHECK IF ABORT FLAG REGISTERED
9034                                                                    ;THIS EVENT AND CHECK IF STATUS REGS
9035                                                                    ;CONTAINED EXPECTED VALUES.
9036                                                                    ;IF NO ABORT OCCURRED THEN GO TO ERROR
9037                                                                    ;OTHERWISE CONTINUE.
9038  042402  012737  077404  177600            MOV     #77404,@#177600 ;SETUP FOR AN ABORT
9039  042410  004767  000152                   JSR     PC,TS7          ;CAUSE AN ABORT TO OCCUR AND
9040                                                                    ;THEN CHECK IF ABORT FLAG REGISTERED
9041                                                                    ;THIS EVENT AND CHECK IF STATUS REGS
9042                                                                    ;CONTAINED EXPECTED VALUES.
9043                                                                    ;IF NO ABORT OCCURRED THEN GO TO ERROR
9044                                                                    ;OTHERWISE CONTINUE.
9045  042414  012701  000220                   MOV     #220,R1         ;
9046  042420  004767  067720                   JSR     PC,MMU          ;INIT MMU
9047  042424  005003                            CLR     R3              ;SETUP MMR1 EXPECTED DATA
9048  042426  012767  000001  140342           MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9049  042434  012737  000001  177572           MOV     #1,@#177572     ;TURN MMU ON
9050  042442  012737  100000  177776           MOV     #100000,@#177776 ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9051  042450  012241                            MOV     (R2)+,-(R1)     ;CAUSE AN ABORT
9052  042452  004767  000220                   JSR     PC,TSM7         ;CHECK IF AN ABORT OCCURRED BY
9053                                                                    ;CHECKING ABORT FLAG AND STATUS REGS
9054                                                                    ;IF NO ABORT OCCURRED THEN GO TO ERROR
9055                                                                    ;OTHERWISE CONTINUE.
9056  042456  005067  140322                   CLR     SAVMR0          ;CLEAR STATUS REGS SAVE AREAS
9057  042462  005067  140320                   CLR     SAVMR1          ;
9058  042466  005067  140316                   CLR     SAVMR2          ;
9059  042472  012703  000022                   MOV     #22,R3          ;SETUP MMR1 EXPECTED DATA
9060  042476  012767  000001  140272           MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9061  042504  012737  000001  177572           MOV     #1,@#177572     ;TURN MMU ON
9062  042512  012737  020000  177776           MOV     #20000,@#177776 ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9063  042520  006522                            MFPI    (R2)+           ;CAUSE AN ABORT
9064  042522  004767  000150                   JSR     PC,TSM7         ;CHECK IF AN ABORT OCCURRED BY
9065                                                                    ;CHECKING ABORT FLAG AND STATUS REGS
9066                                                                    ;IF NO ABORT OCCURRED THEN GO TO ERROR
9067                                                                    ;OTHERWISE CONTINUE.
9068  042526  012737  030000  177776           MOV     #30000,@#177776 ;SETUP PSW
9069  042534  012737  077400  177600           MOV     #77400,@#177600 ;SETUP FOR AN ABORT
9070  042542  005037  177572                   CLR     @#177572        ;MMU OFF
9071  042546  006522                            MFPI    (R2)+           ;TRY TO CAUSE AN ABORT
9072  042550  012603                            MOV     (SP)+,R3        ;POP THE STACK
9073  042552  012637  000216                   MOV     (SP)+,@#216     ;RESTORE DATA AT TEST LOCATIONS
9074  042556  012637  0U0214                   MOV     (SP)+,@#214     ;
9075
9076  042562  000167  000154                   JMP     TS7FIN
9077                                    ;
9078                                    ;ROUTINE TO CAUSE AND CHECK  NONRESIDENT ABORTS
9079                                    ;
9080  042566  012767  000001  140202  TS7:     MOV     #1,FLAG                 ;SETUP FOR AN ABORT
9081  042574  012737  000001  177572           MOV     #1,@#177572             ;TURN MMU ON
9082  042602  010701                            MOV     R7,R1                   ;SAVE PC
9083  042604  006522                            MFPI    (R2)+                   ;CAUSE AN ABORT
9084  042606  J22767  000000  140162           CMP     #0,FLAG                 ;DID AN ABORT OCCUR
9085  042614  001401                            BEQ     OK7                     ;IF YES GO ON
9086  042616  104002                            ERROR   +2              ;MMU ERROR
9087                                                                    ;IF NO GO TO ERROR
```

```
9088  042620  105067  140160          OK7:    CLRB    SAVMR0                  ;SETUP EXPECTED DATA
9089  042624  022767  100000  140152          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
9090  042632  001401                          BEQ     OKA7                    ;IF OK THEN CONTINUE
9091  042634  104002                          ERROR   .+2             ;MMU ERROR
9092                                                                  ;NOT OK THEN GO TO ERROR
9093  042636  026727  140144  000022  OKA7:   CMP     SAVMR1,#22              ; TEST MMR1 FOR EXPECTED VALUE
9094  042644  001401                          BEQ     OKAY7                   ;IF OK THEN CONTINUE
9095  042646  104002                          ERROR   .+2             ;MMU ERROR
9096                                                                  ;NOT OK THEN GO TO ERROR
9097  042650  026701  140134          OKAY7:  CMP     SAVMR2,R1               ; TEST MMR2 FOR EXPECTED VALUE
9098  042654  001401                          BEQ     OKAY7A                  ;IF OK THEN CONTINUE
9099  042656  104002                          ERROR   .+2             ;MMU ERROR
9100                                                                  ;NOT OK THEN GO TO ERROR
9101  042660  005067  140120          OKAY7A: CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9102  042664  005067  140116                  CLR     SAVMR1                  ;
9103  042670  005067  140114                  CLR     SAVMR2                  ;
9104  042674  000207                          RTS     PC                      ;RETURN
9105                                  ;
9106                                  ;ROUTINE TO CHECK IF A NONRESIDENT ABORT OCCURRED
9107                                  ;
9108  042676  022767  000000  140072  TSM7:   CMP     #0,FLAG                 ;DID AN ABORT OCCUR
9109  042704  001401                          BEQ     TSMA                    ;IF YES GO ON
9110  042706  104002                          ERROR   .+2             ;MMU ERROR
9111                                                                  ;IF NO THEN GO TO ERROR
9112  042710  042737  040377  003004  TSMA:   BIC     #40377,@#SAVMR0         ;SETUP EXPECTED DATA
9113  042716  022767  100000  140060          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
9114  042724  001401                          BEQ     TSMB                    ;IF OK THEN CONTINUE
9115  042726  104002                          ERROR   .+2             ;MMU ERROR
9116                                                                  ;IF NO THEN GO TO ERROR
9117  042730  020367  140052          TSMB:   CMP     R3,SAVMR1               ; TEST MMR1 FOR EXPECTED VALUE
9118  042734  001401                          BEQ     TSMC                    ;IF OK THEN CONTINUE
9119  042736  104002                          ERROR   .+2             ;MMU ERROR
9120                                                                  ;IF NOT OK THEN GO TO ERROR
9121  042740  000207                  TSMC:   RTS     PC                      ;RETURN
9122                                  ;
9123  042742  000240                  TS7FIN: NOP
9124  042744                          TSMMU8:
9125                                  ;       TEST READ ONLY ABORTS
9126  042744  005037  177572                  CLR     @#177572                ;MMU OFF
9127  042750  005067  140022                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
9128  042754  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATIONS
9129  042760  013746  000246                  MOV     @#246,-(SP)             ;
9130  042764  005067  140014                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9131  042770  005067  140012                  CLR     SAVMR1                  ;
9132  042774  005067  140010                  CLR     SAVMR2                  ;
9133  043000  004767  067340                  JSR     PC,MMU                  ;INIT MMU
9134  043004  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
9135  043012  012702  000244                  MOV     #244,R2                 ;
9136  043016  012737  077402  177600          MOV     #77402,@#177600         ;SETUP FOR AN ABORT
9137  043024  012746  000246                  MOV     #246,-(SP)              ;PUSH DATA ONTO THE STACK
9138  043030  012767  000001  137740          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
9139  043036  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
9140  043044  010701                          MOV     R7,R1                   ;SAVE PC
9141  043046  006622                          MTPI    (R2)+                   ;CAUSE ABORT
9142  043050  022767  000000  137720          CMP     #0,FLAG                 ;DID ABORT OCCUR
9143  043056  001401                          BEQ     1$                      ;IF YES THEN GO ON
```

```
 9144  043060  104002                          ERROR   +2              ;MMU ERROR
 9145                                                                  ;IF NO THEN GO TO ERROR
 9146  043062  105067  137716          1$:     CLRB    SAVMR0                  ;SETUP EXPECTED DATA
 9147  043066  022767  020000  137710          CMP     #20000,SAVMR0           ; TEST MMR0 FOR EXPECTED VALUE
 9148  043074  001401                          BEQ     2$                      ;IF OK THEN CONTINUE
 9149  043076  104002                          ERROR   +2              ;MMU ERROR
 9150                                                                  ;OTHERWISE GO TO ERROR
 9151  043100  022767  011026  137700  2$:     CMP     #11026,SAVMR1           ; TEST MMR1 FOR EXPECTED VALUE
 9152  043106  001401                          BEQ     3$                      ;IF OK THEN CONTINUE
 9153  043110  104002                          ERROR   +2              ;MMU ERROR
 9154                                                                  ;OTHERWISE GO TO ERROR
 9155  043112  020167  137672          3$:     CMP     R1,SAVMR2               ; TEST MMR2 FOR EXPECTED VALUE
 9156  043116  001401                          BEQ     4$                      ;IF OK THEN CONTINUE
 9157  043120  104002                          ERROR   +2              ;MMU ERROR
 9158                                                                  ;OTHERWISE GO TO ERROR
 9159  043122  012737  030000  177776  4$:     MOV     #30000,@#177776         ;SETUP PSW
 9160  043130  012746  000002                  MOV     #2,-(SP)                ;PUSH DATA ONTO STACK
 9161  043134  006622                          MTPI    (R2)+                   ;TRY TO CAUSE ABORT
 9162  043136  012637  000246                  MOV     (SP)+,@#246             ;RESTORE DATA AT TEST LOCATIONS
 9163  043142  012637  000244                  MOV     (SP)+,@#244             ;
 9164
 9165  043146                          TSMMU9:
 9166                                  ;       TEST PAGE LENGTH ERROR ABORTS
 9167  043146  005037  177572                  CLR     @#177572                ;MMU OFF
 9168  043152  005067  137620                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9169  043156  005067  137622                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9170  043162  005067  137620                  CLR     SAVMR1                  ;
 9171  043166  005067  137616                  CLR     SAVMR2                  ;
 9172  043172  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 9173  043200  004767  067140                  JSR     PC,MMU                  ;INIT MMU
 9174  043204  012703  043466                  MOV     #PLF0,R3                ;LET R3, R1, AND R2 POINT TO THE
 9175  043210  012701  043540                  MOV     #BN0,R1                 ;UPWARD EXPANSION TABLES
 9176  043214  012702  043610                  MOV     #ABORT0,R2              ;
 9177  043220  012737  000026  172516          MOV     #26,@#172516            ;DISABLE USER DATA SPACE
 9178  043226  004767  000050                  JSR     PC,TSM9                 ;TURN MMU ON
 9179                                                                  ;DO RELOCATIONS FOR THE DIFFERENT
 9180                                                                  ;VALUES OF THE PAGE LENGTH FIELD AND
 9181                                                                  ;BLOCK NUMBER.  IF AN ABORT OCCURS
 9182                                                                  ;CHECK TO SEE IF IT WAS SUPPOSED TO,
 9183                                                                  ;AND IF YES CHECK ABORT FLAG AND
 9184                                                                  ;STATUS REGISTERS.
 9185  043232  012703  043660                  MOV     #PLF1,R3                ;LET R3, R1, AND R2 POINT TO THE
 9186  043236  012701  043730                  MOV     #BN1,R1                 ;DOWNWARD EXPANSION TABLES
 9187  043242  012702  043776                  MOV     #ABORT7,R2              ;
 9188  043246  004767  000030                  JSR     PC,TSM9                 ;TURN MMU ON
 9189                                                                  ;DO RELOCATIONS FOR THE DIFFERENT
 9190                                                                  ;VALUES OF THE PAGE LENGTH FIELD AND
 9191                                                                  ;BLOCK NUMBER.  IF AN ABORT OCCURS
 9192                                                                  ;CHECK TO SEE IF IT WAS SUPPOSED TO,
 9193                                                                  ;AND IF YES CHECK ABORT FLAG AND
 9194                                                                  ;STATUS REGISTERS.
 9195  043252  005037  177572                  CLR     @#177572                ;MMU OFF
 9196  043256  012703  043670                  MOV     #PLF1+10,R3             ;POINT TO A VALUE WHICH SHOULD CAUSE
 9197  043262  012701  043740                  MOV     #BN1+10,R1              ;AN ABORT IF MMU IS ON.
 9198  043266  011337  177600                  MOV     (R3),@#177600           ;SETUP UIPDR0
 9199  043272  006521                          MFPI    (R1)+                   ;DO A RELOCATION
```

```
 9200  043274  012605                          MOV     (SP)+,R5                ;POP THE STACK
 9201
 9202  043276  000167  000542                  JMP     TS9FIN
 9203
 9204                                          ;
 9205                                          ;ROUTINE TO CAUSE AND CHECK PAGE LENGTH ERROR ABORTS
 9206  043302  012337  177600          TSM9:   MOV     (R3)+,@#177600          ;SETUP UIPDR0
 9207  043306  010100                          MOV     R1,R0                   ;SAVE A COPY OF R1
 9208  043310  012767  000001  137460          MOV     #1,FLAG                 ;SETUP FOR AN ABORT
 9209  043316  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9210  043324  010704                          MOV     R7,R4                   ;SAVE PC
 9211  043326  006530                          MFPI    @(R0)+                  ;DO A RELOCATION OPERATION
 9212  043330  021227  000000                  CMP     (R2),#0                 ;WAS AN ABORT SUPPOSED TO OCCUR
 9213  043334  001007                          BNE     2$                      ;IF YES GO TO 2$
 9214  043336  012605                          MOV     (SP)+,R5                ;POP THE STACK
 9215  043340  022767  000001  137430          CMP     #1,FLAG                 ;DID AN ABORT OCCUR
 9216  043346  001401                          BEQ     1$                      ;NO GO ON
 9217  043350  104002                          ERROR   .+2                     ;MMU ERROR
 9218                                                                          ;YES GO TO ERROR
 9219  043352  000425                  1$:     BR      6$                      ;
 9220  043354  022767  000000  137414  2$:     CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9221  043362  001401                          BEQ     3$                      ;YES GO ON
 9222  043364  104002                          ERROR   .+2                     ;MMU ERROR
 9223                                                                          ;NO GO TO ERROR
 9224  043366  105067  137412          3$:     CLRB    SAVMR0                  ;SETUP EXPECTED DATA
 9225  043372  022767  040000  137404          CMP     #40000,SAVMR0           ; TEST MMR0 FOR EXPECTED VALUE
 9226  043400  001401                          BEQ     4$                      ;IF OK THEN CONTINUE
 9227  043402  104002                          ERROR   .+2                     ;MMU ERROR
 9228                                                                          ;NOT OK THEN GO TO ERROR
 9229  043404  022767  000020  137374  4$:     CMP     #20,SAVMR1              ; TEST MMR1 FOR EXPECTED VALUE
 9230  043412  001401                          BEQ     5$                      ;IF OK THEN CONTINUE
 9231  043414  104002                          ERROR   .+2                     ;MMU ERROR
 9232                                                                          ;NOT OK THEN GO TO ERROR
 9233  043416  020467  137366          5$:     CMP     R4,SAVMR2               ; TEST MMR2 FOR EXPECTED VALUE
 9234  043422  001401                          BEQ     6$                      ;IF OK THEN CONTINUE
 9235  043424  104002                          ERROR   .+2                     ;MMU ERROR
 9236                                                                          ;NOT OK THEN GO TO ERROR
 9237  043426  005067  137344          6$:     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9238  043432  005067  137346                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9239  043436  005067  137344                  CLR     SAVMR1                  ;
 9240  043442  005067  137342                  CLR     SAVMR2                  ;
 9241  043446  005201                          INC     R1                      ;POINT TO NEXT ENTRY
 9242  043450  005201                          INC     R1                      ;
 9243  043452  005202                          INC     R2                      ;
 9244  043454  005202                          INC     R2                      ;
 9245  043456  021327  000777                  CMP     (R3),#777               ;HAVE ALL ENTRIES BEEN TRIED
 9246  043462  001307                          BNE     TSM9                    ;NO REPEAT
 9247  043464  000207                          RTS     PC                      ;YES RETURN
 9248                                          ;
 9249                                          ;UPWARD EXPANSION TABLES
 9250                                          ;
 9251  043466  070006          PLF0:   .WORD   70006
 9252  043470  070006                  .WORD   70006
 9253  043472  070006                  .WORD   70006
 9254  043474  013406                  .WORD   13406
 9255  043476  020006                  .WORD   20006
```

```
 9256  043500  004006                        .WORD   04006
 9257  043502  040006                        .WORD   40006
 9258  043504  070006                        .WORD   70006
 9259  043506  024006                        .WORD   24006
 9260  043510  004006                        .WORD   04006
 9261  043512  014006                        .WORD   14006
 9262  043514  012006                        .WORD   12006
 9263  043516  002006                        .WORD   02006
 9264  043520  001406                        .WORD   01406
 9265  043522  004006                        .WORD   04006
 9266  043524  002006                        .WORD   02006
 9267  043526  000406                        .WORD   00406
 9268  043530  007406                        .WORD   07406
 9269  043532  001006                        .WORD   01006
 9270  043534  003406                        .WORD   03406
 9271  043536  000777                        .WORD   777
 9272  043540  013000                BNO:    .WORD   013000
 9273  043542  016000                        .WORD   016000
 9274  043544  017000                        .WORD   017000
 9275  043546  002700                        .WORD   002700
 9276  043550  014000                        .WORD   014000
 9277  043552  002000                        .WORD   002000
 9278  043554  004000                        .WORD   004000
 9279  043556  007000                        .WORD   007000
 9280  043560  002000                        .WORD   002000
 9281  043562  000700                        .WORD   000700
 9282  043564  004000                        .WORD   004000
 9283  043566  001000                        .WORD   001000
 9284  043570  000300                        .WORD   000300
 9285  043572  000400                        .WORD   000400
 9286  043574  001400                        .WORD   001400
 9287  043576  000600                        .WORD   000600
 9288  043600  000200                        .WORD   000200
 9289  043602  001700                        .WORD   001700
 9290  043604  000300                        .WORD   000300
 9291  043606  000700                        .WORD   000700
 9292  043610  000000                ABORTO: .WORD   0
 9293  043612  000000                        .WORD   0
 9294  043614  000001                        .WORD   1
 9295  043616  000000                        .WORD   0
 9296  043620  000001                        .WORD   1
 9297  043622  000001                        .WORD   1
 9298  043624  000000                        .WORD   0
 9299  043626  000000                        .WORD   0
 9300  043630  000000                        .WORD   0
 9301  043632  000000                        .WORD   0
 9302  043634  000001                        .WORD   1
 9303  043636  000000                        .WORD   0
 9304  043640  000000                        .WORD   0
 9305  043642  000001                        .WORD   1
 9306  043644  000001                        .WORD   1
 9307  043646  000001                        .WORD   1
 9308  043650  000001                        .WORD   1
 9309  043652  000000                        .WORD   0
 9310  043654  000001                        .WORD   1
 9311  043656  000000                        .WORD   0
```

```
                              ;
                              ;DOWNWARD EXPANSION TABLES
                              ;
043660  000416                PLF1:    .WORD   00416
043662  020016                         .WORD   20016
043664  024016                         .WORD   24016
043666  034016                         .WORD   34016
043670  074016                         .WORD   74016
043672  040016                         .WORD   40016
043674  020016                         .WORD   20016
043676  000016                         .WORD   00016
043700  030016                         .WORD   30016
043702  010016                         .WORD   10016
043704  014016                         .WORD   14016
043706  004016                         .WORD   04016
043710  002016                         .WORD   02016
043712  000416                         .WORD   00416
043714  000016                         .WORD   00016
043716  003416                         .WORD   03416
043720  001016                         .WORD   01016
043722  001416                         .WORD   01416
043724  000416                         .WORD   00416
043726  000777                         .WORD   777
043730  000100                BN1:     .WORD   000100
043732  010000                         .WORD   010000
043734  006000                         .WORD   006000
043736  016000                         .WORD   016000
043740  016000                         .WORD   016000
043742  004000                         .WORD   004000
043744  000000                         .WORD   000000
043746  000000                         .WORD   000000
043750  004000                         .WORD   004000
043752  004000                         .WORD   004000
043754  004000                         .WORD   004000
043756  000000                         .WORD   000000
043760  000300                         .WORD   000300
043762  000000                         .WORD   000000
043764  000400                         .WORD   000400
043766  001000                         .WORD   001000
043770  000100                         .WORD   000100
043772  000400                         .WORD   000400
043774  000200                         .WORD   000200
043776  000000                ABORT7:  .WORD   0
044000  000000                         .WORD   0
044002  000000                         .WORD   0
044004  000000                         .WORD   0
044006  000001                         .WORD   1
044010  000001                         .WORD   1
044012  000001                         .WORD   1
044014  000000                         .WORD   0
044016  000001                         .WORD   1
044020  000000                         .WORD   0
044022  000000                         .WORD   0
044024  000001                         .WORD   1
044026  000001                         .WORD   1
044030  000001                         .WORD   1
```

```
 9368  044032  000000                          .WORD   0
 9369  044034  000000                          .WORD   0
 9370  044036  000001                          .WORD   1
 9371  044040  000000                          .WORD   0
 9372  044042  000000                          .WORD   0
 9373                                  ;
 9374  044044  000240                  TS9FIN: NOP
 9375  044046                          TSMM10:
 9376                                  ;       FUNCTIONAL TEST OF BITS <6:1> OF MMR0
 9377  044046  005037  177572                  CLR     @#177572                ;MMU OFF
 9378  044052  005067  136720                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9379  044056  005067  136722                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9380  044062  005067  136720                  CLR     SAVMR1                  ;
 9381  044066  005067  136716                  CLR     SAVMR2                  ;
 9382  044072  004767  066246                  JSR     PC,MMU                  ;INIT MMU
 9383  044076  005037  177776                  CLR     @#177776                ;INIT PSW: PREVIOUS MODE = KERNAL
 9384  044102  012702  020200                  MOV     #20200,R2               ;
 9385  044106  012737  077400  172302          MOV     #77400,@#172302         ;SETUP KIPDR1 TO ABORT
 9386  044114  012767  000001  136654          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9387  044122  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9388  044130  010701                          MOV     R7,R1                   ;SAVE PC
 9389  044132  006522                          MFPI    (R2)+                   ;DO A RELOCATION VIA KIPAR1
 9390  044134  012704  100003                  MOV     #100003,R4              ;SETUP EXPECTED DATA
 9391  044140  004767  000202                  JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9392                                                                          ;IF YES CHECK BITS <6:1> OF MMR0.
 9393  044144  012737  030000  177776          MOV     #30000,@#177776         ;INIT PSW: PREVIOUS MODE = USER
 9394  044152  004767  066166                  JSR     PC,MMU                  ;INIT MMU
 9395  044156  012737  077400  177636          MOV     #77400,@#177636         ;SETUP UDPDR7 TO ABORT
 9396  044164  012702  160000                  MOV     #160000,R2              ;
 9397  044170  012767  000001  136600          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9398  044176  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9399  044204  010701                          MOV     R7,R1                   ;SAVE PC
 9400  044206  106522                          MFPD    (R2)+                   ;DO A RELOCATION VIA UDPAR7
 9401  044210  012704  100177                  MOV     #100177,R4              ;SETUP EXPECTED DATA
 9402  044214  004767  000126                  JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9403                                                                          ;IF YES CHECK BITS <6:1> OF MMR0.
 9404  044220  012737  010000  177776          MOV     #10000,@#177776         ;INIT PSW: PREVIOUS MODE= SUPERVISOR
 9405  044226  004767  066112                  JSR     PC,MMU                  ;INIT MMU
 9406  044232  012737  077400  172212          MOV     #77400,@#172212         ;SETUP SIPDR5 TO ABORT
 9407  044240  012702  120000                  MOV     #120000,R2              ;ACCESS PAGE 05
 9408  044244  012767  000001  136524          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9409  044252  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9410  044260  010701                          MOV     R7,R1                   ;SAVE PC
 9411  044262  006522                          MFPI    (R2)+                   ;DO A RELOCATION VIA SIPAR5
 9412  044264  012704  100053                  MOV     #100053,R4              ;SETUP EXPECTED DATA:ABORT, PAGE 05
 9413  044270  004767  000052                  JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9414                                                                          ;IF YES CHECK BITS <6:1> OF MMR0.
 9415                                  ;
 9416                                  ;TEST THAT ILLEGAL MODE CAUSES MMU ABORT
 9417                                  ;
 9418  044274  012737  020000  177776          MOV     #20000,@#177776         ;INIT PSW:SET ILLEGAL PREVIOUS MODE
 9419  044302  004767  066036                  JSR     PC,MMU                  ;INIT MMU
 9420  044306  012702  040000                  MOV     #40000,R2               ;SET UP ACCESS TO PAGE 2
 9421  044312  012767  000001  136456          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9422  044320  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9423  044326  010701                          MOV     R7,R1                   ;SAVE PC
```

```
 9424  044330  106522                          MFPD    (R2)+                   ;DO A RELOCATION
 9425  044332  012704  100105                  MOV     #100105,R4              ;SETUP EXPECTED DATA:ABORT, ILLEGAL
 9426                                                                          ; PROCESSOR MODE, PAGE 02
 9427  044336  004767  000004                  JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9428                                                                          ;IF YES CHECK BITS <6:1> OF MMR0.
 9429
 9430  044342  000167  000062                  JMP     T10FIN
 9431                                  ;
 9432                                  ;ROUTINE TO CHECK IF A MMU ABORT OCCURRED AND IF STATUS REG MMR0
 9433                                  ;CONTAINS EXPECTED DATA
 9434                                  ;
 9435  044346  022767  000000  136422  TS10:   CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9436  044354  001401                          BEQ     1$                      ;YES GO ON
 9437  044356  104002                          ERROR   .2              ;MMU ERROR
 9438                                                                  ;NO GO TO ERROR
 9439  044360  020467  136420          1$:     CMP     R4,SAVMR0               ; TEST MMR0 FOR EXPECTED DATA
 9440  044364  001401                          BEQ     2$                      ;OK GO ON
 9441  044366  104002                          ERROR   .2              ;MMU ERROR
 9442                                                                  ;NO GO TO ERROR
 9443  044370  022767  000022  136410  2$:     CMP     #22,SAVMR1              ; TEST MMR1 FOR EXPECTED DATA
 9444  044376  001401                          BEQ     3$                      ;OK GO ON
 9445  044400  104002                          ERROR   .2              ;MMU ERROR
 9446                                                                  ;NO GO TO ERROR
 9447  044402  020167  136402          3$:     CMP     R1,SAVMR2               ; TEST MMR2 FOR EXPECTED DATA
 9448  044406  001401                          BEQ     4$                      ;OK GO ON
 9449  044410  104002                          ERROR   .2              ;MMU ERROR
 9450                                                                  ;NO GO TO ERROR
 9451  044412  005067  136366          4$:     CLR     SAVMR0                  ;CLEAR MMU STATUS REGS SAVE AREAS
 9452  044416  005067  136364                  CLR     SAVMR1                  ;
 9453  044422  005067  136362                  CLR     SAVMR2                  ;
 9454  044426  000207                          RTS     PC                      ;RETURN
 9455
 9456  044430                          T10FIN:
 9457  044430                          TSMM11:
 9458                                  ;       TEST DATA SPACE BITS MMR3
 9459  044430  005037  177572                  CLR     @#177572                ;MMU OFF
 9460  044434  005067  136336                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9461  044440  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 9462  044446  012701  000026                  MOV     #26,R1                  ;SETUP FIRST MMR3 VALUE
 9463  044452  012703  177610                  MOV     #177610,R3              ;POINT TO UIPDR4
 9464  044456  012704  000021                  MOV     #21,R4                  ;SETUP SECOND MMR3 VALUE
 9465  044462  004767  000060                  JSR     PC,TS11                 ; TEST ENABLE USER DATA SPACE BIT
 9466  044466  012737  000000  177776          MOV     #0,@#177776             ;SETUP PSW
 9467  044474  012701  000023                  MOV     #23,R1                  ;SETUP FIRST MMR3 VALUE
 9468  044500  012703  172310                  MOV     #172310,R3              ;POINT TO KIPDR4
 9469  044504  012704  000024                  MOV     #24,R4                  ;SETUP SECOND MMR3 VALUE
 9470  044510  004767  000032                  JSR     PC,TS11                 ; TEST ENABLE KERNEL DATA SPACE BIT
 9471  044514  012737  010000  177776          MOV     #10000,@#177776         ;SETUP PSW
 9472  044522  012701  000025                  MOV     #25,R1                  ;SETUP FIRST MMR3 VALUE
 9473  044526  012703  172210                  MOV     #172210,R3              ;POINT TO SIPDR4
 9474  044532  012704  000022                  MOV     #22,R4                  ;SETUP SECOND MMR3 VALUE
 9475  044536  004767  000004                  JSR     PC,TS11                 ; TEST ENABLE SUPERVISOR DATA SPACE BIT
 9476
 9477  044542  000167  000120                  JMP     T11FIN
 9478                                  ;
 9479                                  ;ROUTINE TO TEST ENABLE DATA SPACE BITS OF MMR3
```

```
9480                                    ;
9481  044546  004767  065572            TS11:   JSR     PC,MMU          ;INIT MMU
9482  044552  010137  172516                    MOV     R1,@#172516     ;DISABLE DATA SPACE OF MODE UNDER TEST
9483  044556  012713  077400                    MOV     #77400,(R3)     ;SETUP IPDR TO ABORT
9484  044562  012702  100000                    MOV     #100000,R2      ;
9485  044566  012767  000001  136202            MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9486  044574  012737  000001  177572            MOV     #1,@#177572     ;MMU ON
9487  044602  106522                            MFPD    (R2)+           ;DO A RELOCATION
9488  044604  022767  000000  136164            CMP     #0,FLAG         ;DID AN ABORT OCCUR
9489  044612  001401                            BEQ     1$              ;YES GO ON
9490  044614  104002                            ERROR   +2              ;MMU ERROR
9491                                            ;NO GO TO ERROR
9492  044616  010437  172516            1$:     MOV     R4,@#172516     ;ENABLE DATA SPACE OF MODE UNDER TEST
9493  044622  012702  100000                    MOV     #100000,R2      ;
9494  044626  012767  000001  136142            MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9495  044634  012737  000001  177572            MOV     #1,@#177572     ;MMU ON
9496  044642  106522                            MFPD    (R2)+           ;DO A RELOCATION
9497  044644  005726                            TST     (SP)+           ;POP THE STACK
9498  044646  022767  000001  136122            CMP     #1,FLAG         ;DID AN ABORT OCCUR
9499  044654  001401                            BEQ     2$              ;NO GO ON
9500  044656  104002                            ERROR   +2              ;MMU ERROR
9501                                            ;YES GO TO ERROR
9502  044660  005067  136112            2$:     CLR     FLAG            ;CLEAR MMU ABORT FLAG
9503  044664  000207                            RTS     PC              ;RETURN
9504                                    ;
9505  044666                            T11FIN:
9506  044666                            TSMM12:
9507                                    ;       MMR1 FUNCTIONAL TEST
9508  044666  005037  177572                    CLR     @#177572        ;MMU OFF
9509  044672  005067  136100                    CLR     FLAG            ;CLEAR MMU ABORT FLAG
9510  044676  005067  136104                    CLR     SAVMR1          ;CLEAR STATUS REG SAVE AREA
9511  044702  004767  065436                    JSR     PC,MMU          ;INIT MMU
9512  044706  012737  030000  177776            MOV     #30000,@#177776 ;INIT PSW
9513  044714  012704  100200                    MOV     #100200,R4      ;SETUP TEST LOCATIONS
9514  044720  010401                            MOV     R4,R1           ;
9515  044722  012705  100101                    MOV     #100101,R5      ;
9516  044726  010502                            MOV     R5,R2           ;
9517  044730  012737  000020  172516            MOV     #20,@#172516    ;INIT MMR3
9518  044736  012737  077402  172310            MOV     #77402,@#172310 ;SETUP KIPDR4 TO ABORT
9519  044744  012703  006414                    MOV     #6414,R3        ;SETUP EXPECTED DATA FOR MMR1
9520  044750  012767  000001  136020            MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9521  044756  012737  000001  177572            MOV     #1,@#177572     ;TURN MMU ON
9522  044764  010767  135772                    MOV     R7,SLOC00       ;SAVE PC
9523  044770  112425                            MOVB    (R4)+,(R5)+     ;DO A RELOCATION
9524  044772  004767  000206                    JSR     PC,TS12         ;CHECK IF AN ABORT OCCURRED AND IF
9525                                            ;YES IF MMR1 EQUALS EXPECTED DATA
9526  044776  012703  175011                    MOV     #175011,R3      ;SETUP EXPECTED DATA FOR MMR1
9527  045002  012767  000001  135766            MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9528  045010  012737  000001  177572            MOV     #1,@#177572     ;TURN MMU ON
9529  045016  010767  135740                    MOV     R7,SLOC00       ;SAVE PC
9530  045022  112142                            MOVB    (R1)+,-(R2)     ;DO A RELOCATION
9531  045024  004767  000154                    JSR     PC,TS12         ;CHECK IF AN ABORT OCCURRED AND IF
9532                                            ;YES IF MMR1 EQUALS EXPECTED DATA
9533  045030  012703  006771                    MOV     #6771,R3        ;SETUP EXPECTED DATA FOR MMR1
9534  045034  012767  000001  135734            MOV     #1,FLAG         ;SETUP FLAG FOR AN ABORT
9535  045042  012737  000001  177572            MOV     #1,@#177572     ;TURN MMU ON
```

```
 9536  045050  010767  135706          MOV     R7,SLOC00               ;SAVE PC
 9537  045054  114125                  MOVB    -(R1),(R5)+             ;DO A RELOCATION
 9538  045056  004767  000122          JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
 9539                                                                  ;YES IF MMR1 EQUALS EXPECTED DATA
 9540  045062  012703  006411          MOV     #6411,R3                ;SETUP EXPECTED DATA FOR MMR1
 9541  045066  012767  000001  135702  MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9542  045074  012737  000001  177572  MOV     #1,@#177572             ;TURN MMU ON
 9543  045102  010767  135654          MOV     R7,SLOC00               ;SAVE PC
 9544  045106  112125                  MOVB    (R1)+,(R5)+             ;DO A RELOCATION
 9545  045110  004767  000070          JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
 9546                                                                  ;YES IF MMR1 EQUALS EXPECTED DATA
 9547  045114  012703  171025          MOV     #171025,R3              ;SETUP EXPECTED DATA FOR MMR1
 9548  045120  012767  000001  135650  MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9549  045126  012737  000001  177572  MOV     #1,@#177572             ;TURN MMU ON
 9550  045134  010767  135622          MOV     R7,SLOC00               ;SAVE PC
 9551  045140  012542                  MOV     (R5)+,-(R2)             ;DO A RELOCATION
 9552  045142  004767  000036          JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
 9553                                                                  ;YES IF MMR1 EQUALS EXPECTED DATA
 9554  045146  012703  012762          MOV     #12762,R3               ;SETUP EXPECTED DATA FOR MMR1
 9555  045152  012767  000001  135616  MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9556  045160  012737  000001  177572  MOV     #1,@#177572             ;TURN MMU ON
 9557  045166  010767  135570          MOV     R7,SLOC00               ;SAVE PC
 9558  045172  014225                  MOV     -(R2),(R5)+             ;DO A RELOCATION
 9559  045174  004767  000004          JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
 9560                                                                  ;YES IF MMR1 EQUALS EXPECTED DATA
 9561
 9562  045200  000167  000046          JMP     T12FIN
 9563                                  ;
 9564                                  ;ROUTINE TO CHECK IF AN ABORT OCCURRED AND IF MMR1 EQUALS EXPECTED DATA
 9565                                  ;
 9566  045204  022767  000000  135564  TS12:   CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9567  045212  001401                          BEQ     1$                      ;YES GO ON
 9568  045214  104002                          ERROR   .+2             ;MMU ERROR
 9569                                                                  ;NO GO TO ERROR
 9570  045216  020367  135564          1$:     CMP     R3,SAVMR1               ; TEST MMR1 FOR EXPECTED DATA
 9571  045222  001401                          BEQ     2$                      ;OK GO ON
 9572  045224  104002                          ERROR   .+2             ;MMU ERROR
 9573                                                                  ;NO GO TO ERROR
 9574  045226  026767  135530  135554  2$:     CMP     SLOC00,SAVMR2           ; TEST MMR2 FOR EXPECTED DATA
 9575  045234  001401                          BEQ     3$                      ;OK GO ON
 9576  045236  104002                          ERROR   .+2             ;MMU ERROR
 9577                                                                  ;NO GO TO ERROR
 9578  045240  005067  135542          3$:     CLR     SAVMR1                  ;CLEAR STATUS REG SAVE AREA
 9579  045244  005067  135540                  CLR     SAVMR2                  ;
 9580  045250  000207                          RTS     PC                      ;RETURN
 9581                                  ;
 9582  045252  000240                  T12FIN: NOP
 9583  045254                          TSMM13:
 9584                                  ;       ADDER RELOCATION TEST PART A
 9585                                  ;;**************************************************************
 9586                                  ;(NEED 16 BITS OF MEMORY ADDRESSING)
 9587  045254  005037  177572                  CLR     @#177572                ;MMU OFF
 9588  045260  005067  135512                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9589  045264  005037  177776                  CLR     @#177776                ;INIT PSW
 9590  045270  004767  065050                  JSR     PC,MMU                  ;INIT MMU
 9591  045274  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
```

```
9592  045302  012703  045456          MOV     #PARAD1,R3              ;SETUP PARS WITH TEST VALUES
9593  045306  012701  045510          MOV     #PARVA1,R1              ;
9594  045312  012133          1$:     MOV     (R1)+,@(R3)+            ;
9595  045314  021127  000333          CMP     (R1),#333               ;
9596  045320  001374                  BNE     1$                      ;
9597  045322  012703  045574          MOV     #PHY1,R3                ;SET POINTERS TO ADDER PART A
9598  045326  012701  045542          MOV     #VIR1,R1                ; TEST TABLES.
9599  045332  012702  045626          MOV     #MODE1,R2               ;
9600  045336  012237  177776      2$: MOV     (R2)+,@#177776          ;INIT PSW
9601  045342  013305                  MOV     @(R3)+,R5               ;SAVE DATA AT PHYSICAL ADDRESS
9602  045344  012737  000001  177572  MOV     #1,@#177572             ;TURN MMU ON
9603  045352  006531                  MFPI    @(R1)+                  ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9604  045354  012604                  MOV     (SP)+,R4                ;
9605  045356  005037  177572          CLR     @#177572                ;TURN MMU OFF
9606  045362  020504                  CMP     R5,R4                   ;IS DATA EQUAL TO EXPECTED
9607  045364  001401                  BEQ     3$                      ;YES GO ON
9608  045366  104002                  ERROR   +2              ;MMU ERROR
9609                                                          ;NO IT IS AN ADDER ERROR
9610  045370  021327  000111      3$: CMP     (R3),#111               ;ARE WE READY TO TEST DATA SPACE
9611  045374  001360                  BNE     2$                      ;NO GO TO 2$
9612  045376  005203                  INC     R3                      ;POINT TO DATA SPACE VALUES
9613  045400  005203                  INC     R3                      ;
9614  045402  005201                  INC     R1                      ;
9615  045404  005201                  INC     R1                      ;
9616  045406  005202                  INC     R2                      ;
9617  045410  005202                  INC     R2                      ;
9618  045412  012237  177776          MOV     (R2)+,@#177776          ;INIT PSW
9619  045416  013305                  MOV     @(R3)+,R5               ;SAVE DATA AT PHYSICAL ADDRESS
9620  045420  012737  000027  172516  MOV     #27,@#172516            ;INIT MMR3
9621  045426  012737  000001  177572  MOV     #1,@#177572             ;TURN MMU ON
9622  045434  106531                  MFPD    @(R1)+                  ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9623  045436  012604                  MOV     (SP)+,R4                ;POP THE STACK
9624  045440  005037  177572          CLR     @#177572                ;TURN MMU OFF
9625  045444  020504                  CMP     R5,R4                   ;IS DATA EQUAL TO EXPECTED
9626  045446  001401                  BEQ     4$                      ;YES GO ON
9627  045450  104002                  ERROR   +2              ;MMU ERROR
9628                                                          ;NO IT IS AN ADDER ERROR
9629  045452                  4$:
9630  045452  000167  000202          JMP     T13FIN
9631                          ;
9632                          ;ADDER TEST PART A TABLES
9633                          ;
9634  045456  172240          PARAD1: .WORD   172240
9635  045460  177642                  .WORD   177642
9636  045462  172252                  .WORD   172252
9637  045464  177640                  .WORD   177640
9638  045466  172242                  .WORD   172242
9639  045470  172254                  .WORD   172254
9640  045472  177652                  .WORD   177652
9641  045474  177644                  .WORD   177644
9642  045476  172246                  .WORD   172246
9643  045500  177654                  .WORD   177654
9644  045502  172250                  .WORD   172250
9645  045504  177660                  .WORD   177660
9646  045506  000333                  .WORD   333
9647  045510  000000          PARVA1: .WORD   000000
```

```
 9648  045512  000010                          .WORD   000010
 9649  045514  177777                          .WORD   177777
 9650  045516  177601                          .WORD   177601
 9651  045520  000010                          .WORD   000010
 9652  045522  000052                          .WORD   000052
 9653  045524  000070                          .WORD   000070
 9654  045526  000010                          .WORD   000010
 9655  045530  000010                          .WORD   000010
 9656  045532  000060                          .WORD   000060
 9657  045534  000000                          .WORD   000000
 9658  045536  000010                          .WORD   000010
 9659  045540  000333                          .WORD   333
 9660  045542  000000                  VIR1:   .WORD   000000
 9661  045544  025000                          .WORD   025000
 9662  045546  135224                          .WORD   135224
 9663  045550  017700                          .WORD   017700
 9664  045552  033000                          .WORD   033000
 9665  045554  145252                          .WORD   145252
 9666  045556  121000                          .WORD   121000
 9667  045560  043000                          .WORD   043000
 9668  045562  075000                          .WORD   075000
 9669  045564  142000                          .WORD   142000
 9670  045566  117700                          .WORD   117700
 9671  045570  000111                          .WORD   111
 9672  045572  007000                          .WORD   007000
 9673  045574  000000                  PHY1:   .WORD   000000
 9674  045576  006000                          .WORD   006000
 9675  045600  015124                          .WORD   015124
 9676  045602  000000                          .WORD   000000
 9677  045604  014000                          .WORD   014000
 9678  045606  012452                          .WORD   012452
 9679  045610  010000                          .WORD   010000
 9680  045612  004000                          .WORD   004000
 9681  045614  016000                          .WORD   016000
 9682  045616  010000                          .WORD   010000
 9683  045620  017700                          .WORD   017700
 9684  045622  000111                          .WORD   111
 9685  045624  010000                          .WORD   010000
 9686  045626  010000                  MODE1:  .WORD   010000
 9687  045630  030000                          .WORD   030000
 9688  045632  010000                          .WORD   010000
 9689  045634  030000                          .WORD   030000
 9690  045636  010000                          .WORD   010000
 9691  045640  010000                          .WORD   010000
 9692  045642  030000                          .WORD   030000
 9693  045644  030000                          .WORD   030000
 9694  045646  010000                          .WORD   010000
 9695  045650  030000                          .WORD   030000
 9696  045652  010000                          .WORD   010000
 9697  045654  000111                          .WORD   111
 9698  045656  030000                          .WORD   030000
 9699                                  ;
 9700  045660                          T13FIN:
 9701  045660                          TS1822:
 9702                                  ;       TEST 22/18 BIT ADDRESS OPTION
 9703                                  ;;*****************************************************************
```

```
                                        ;CHECK THE SOFTWARE SWITCH REGISTER TO DETERMINE IF THIS IS A 22 BIT OR AN
                                        ;18 BIT ADDRESS SYSTEM. BIT 08 IN THE SWR=1 INDICATES AN 18 BIT SYSTEM.
                                        ;IF WE'RE IN A 22 BIT SYSTEM WE CAN PERFORM SOME EXTRA TESTS.
045660  032777  000400  133252          BIT     #BIT08,@SWR             ;IS BIT 08 SET?
045666  001405                          BEQ     100$                    ;BRANCH IF ITS NOT
045670  062737  000001  001204          ADD     #1,@#TESTN              ;KEEP TEST NUMBERS IN ORDER
                                                                        ;ADD 1 FOR THE TESTS WE'RE SKIPPING
045676  000167  001422                  JMP     T14FIN                  ;SKIP OVER THESE TESTS IF IT IS
                                        ;
                                        ;IF THIS IS A 22 BIT SYSTEM CHECK THE 22/18 BIT ADDRESS OPTION
                                        ;
                                        ;TO TEST 22 BIT ADDRESSING WE DO THE FOLLOWING:
                                        ;       A. ENABLE 22 BIT ADDRESSING MODE
                                        ;       B. CLEAR ADDRESS 0
                                        ;       C. WRITE PHYSICAL ADDRESS 17000000 WITH ALL ONES
                                        ;       D. CHECK ADDRESS 0
                                        ;IF ADDRESS 0 IS UNCHANGED (=0) OR A TIME OUT OCCURRED, IT INDICATES
                                        ;22 BIT MODE IS FUNCTIONING.
                                        ;IF ADDRESS 0 =177777 IT INDICATES THAT 22 BIT MODE IS NOT FUNCTIONING.

045702  013767  000004  135052  100$:   MOV     @#4,SLOC00              ;SAVE VECTOR
045710  013767  000006  135046          MOV     @#6,SLOC01              ;SAVE VECTOR
045716  012737  045776  000004          MOV     #1$,@#4                 ;SET VECTOR FOR NXM TRAP
045724  012737  000340  000006          MOV     #340,@#6                ;
045732  012767  000020  124556          MOV     #20,SR3                 ;ENABLE 22 BIT MODE ADDRESSING
045740  012767  170000  124406          MOV     #170000,KIPAR6          ;SET KIPAR6 FOR 1920-1924KW ADDR RANGE
045746  012767  000001  131616          MOV     #1,SR0                  ;ENABLE MMU
045754  005067  132020                  CLR     0                       ;CLEAR ADDR 0
045760  012737  177777  140000          MOV     #177777,@#140000        ;MOVE ALL ONES TO ADDR 17000000 VIA KIPAR6
                                                                        ;A TIME OUT ERROR OR
045766  005767  132006                  TST     0                       ;ADDR 0 REMAINING CLEAR INDICATES
                                                                        ;THAT 22 BIT ADDRESS MODE IS WORKING AND
                                                                        ;THAT SOME FURTHER TESTS SHOULD BE PERFORMED
045772  001405                          BEQ     2$                      ;IF ADDR 0 =177777
                                                                        ;ERROR! 22 BIT ADDRESS MODE BAD
045774  104002                          ERROR   .+2                     ;MMU ERROR
045776  012706  001000          1$:     MOV     #STBOT,SP               ;GOT HERE AS A RESULT OF NXM TRAP---
                                                                        ;CLEAN UP THE STACK
046002  005037  177766                  CLR     @#177766                ;CLEAR CPU ERROR REGISTER
046006  012737  046046  000004  2$:     MOV     #3$,@#4                 ;SET UP VECTOR FOR NXM TRAP
046014  042767  000020  124474          BIC     #BIT04,SR3              ;SET 18 BIT ADDRESSING MODE IN SR3
046022  012767  170000  124324          MOV     #170000,KIPAR6          ;SET KIPAR6 SO THAT BITS 18-21 SHOULD
                                                                        ;BE ASSERTED IF 22 BIT ADR WAS ENABLED
046030  012737  177777  140000          MOV     #177777,@#140000        ;TRY TO WRITE ADDR 17000000 VIA KIPAR6
                                                                        ;ADDR 0 SHOULD = 177777. A TIME OUT
046036  022737  177777  000000          CMP     #177777,@#0             ;OR ADDR 0 = ZERO INDICATES AN ERROR
046044  001403                          BEQ     4$                      ;GO TO NEXT TEST IF ADDR 0=177777
046046  005067  131520          3$:     CLR     SR0                     ;DISABLE MMU BEFORE ERROR.
                                                                        ;ERROR! 18 BIT ADDR OPTION IS N.G.
046052  104002                          ERROR   .+2                     ;MMU ERROR
                                        ;
                                        ;TEST ADDRESS BITS 18 THRU 21
                                        ;
046054  052767  000020  124434  4$:     BIS     #BIT04,SR3              ;ENABLE 22 BIT ADDRESSING MODE
046062  012767  046124  131714          MOV     #5$,4                   ;SET UP FOR NXM TRAP
```

```
 9760  046070  005067  131704                    CLR     0                       ;CLEAR ADDRESS 0
 9761  046074  012767  010000  124252            MOV     #10000,KIPAR6           ;TEST ADDRESS BIT 18
 9762  046102  012737  177777  140000            MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 1000000
 9763  046110  005767  131664                    TST     0                       ;TEST ADDRESS 0. SHOULD = ZERO
 9764  046114  001403                            BEQ     5$                      ;BRANCH IF ADDRESS 0=0
 9765  046116  005067  131450                    CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9766  046122  104002                            ERROR   +2                  ;MMU ERROR
 9767                                                                        ;ERROR! BIT 18 DID NOT ASSERT
 9768  046124  012737  046166  000004  5$:       MOV     #6$,@#4                 ;SET UP FOR NXM TRAP
 9769  046132  005067  131642                    CLR     0                       ;CLEAR ADDR 0
 9770  046136  012767  020000  124210            MOV     #20000,KIPAR6           ;TEST ADDRESS BIT 19
 9771  046144  012737  177777  140000            MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 2000000
 9772  046152  005767  131622                    TST     0                       ;TEST ADDR 0. SHOULD= ZERO
 9773  046156  001403                            BEQ     6$                      ;BRANCH IF ADDRESS 0=0
 9774  046160  005067  131406                    CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9775  046164  104002                            ERROR   +2                  ;MMU ERROR
 9776                                                                        ;ERROR! BIT 19 DID NOT ASSERT
 9777  046166  012737  046230  000004  6$:       MOV     #7$,@#4                 ;SET UP FOR NXM TRAP
 9778  046174  005067  131600                    CLR     0                       ;CLEAR ADDR 0
 9779  046200  012767  040000  124146            MOV     #40000,KIPAR6           ;TEST ADDRESS BIT 20
 9780  046206  012737  177777  140000            MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 4000000
 9781  046214  005767  131560                    TST     0                       ;TEST ADDR 0. SHOULD =0
 9782  046220  001403                            BEQ     7$                      ;BRANCH IF ADDRESS 0 =0
 9783  046222  005067  131344                    CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9784  046226  104002                            ERROR   ,2                  ;MMU ERROR
 9785                                                                        ;ERROR! BIT 20 DID NOT ASSERT
 9786  046230  012737  046272  000004  7$:       MOV     #8$,@#4                 ;SET UP FOR NXM
 9787  046236  005067  131536                    CLR     0                       ;CLEAR ADDRESS 0
 9788  046242  012767  100000  124104            MOV     #100000,KIPAR6          ;TEST ADDRESS BIT 21
 9789  046250  012737  177777  140000            MOV     #177777,@#140000        ;WRITE ALL ONES AT ADDR 10000000
 9790  046256  005767  131516                    TST     0                       ;CHECK ADDRESS 0. SHOULD = 0
 9791  046262  001403                            BEQ     8$                      ;BRANCH IF ADDR 0 = 0
 9792  046264  005067  131302                    CLR     SR0                     ;DISABLE MMU    BEFORE ERROR
 9793  046270  104002                            ERROR   +2                  ;MMU ERROR
 9794                                                                        ;ERROR! ADDR BIT 21 DID NOT ASSERT
 9795  046272  005067  131274            8$:     CLR     SR0                     ;DISABLE MMU
 9796  046276  005037  177766                    CLR     @#177766                ;CLEAR CPU ERROR REGISTER
 9797  046302  012706  001000                    MOV     #STBOT,R6               ;RESET STACK POINTER
 9798  046306  013737  002762  000004            MOV     @#SLOC00,@#4            ;RESTORE VECTORS
 9799  046314  013737  002764  000006            MOV     @#SLOC01,@#6            ;
 9800
 9801  046322                            TSMM14:
 9802                                    ;       ADDER RELOCATION TEST PART B
 9803                                    ;;**************************************************************
 9804                                    ;(NEED 22 BITS OF MEMORY ADDRESSING)
 9805  046322  005037  177572                    CLR     @#SR0                   ;TURN OFF MMU.
 9806  046326  005067  131434                    CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
 9807  046332  013737  000004  002762            MOV     @#4,@#SLOC00            ;SAVE LOC 4 IN SLOC00.
 9808  046340  013737  000006  002764            MOV     @#6,@#SLOC01            ;SAVE LOC 6 IN SLOC01.
 9809  046346  012737  047036  000004            MOV     #NXMTRP,@#4             ;SET UP FOR TIMEOUT TRAP
 9810  046354  012737  000340  000006            MOV     #340,@#6                ;SET UP FOR TIMEOUT TRAP
 9811  046362  005037  172340                    CLR     @#KIPAR0                ;SET KER PAR0 FOR 1ST 4KW OF MEMORY.
 9812  046366  012767  077406  123704            MOV     #77406,KIPDR0           ;SET KER PDR FOR 4KW R/W ACCESS.
 9813  046374  012737  177500  172354            MOV     #177500,@#KIPAR6        ;SET UP KERNEL PAGE ADDR REG 6
 9814                                                                            ;FOR HIGHEST 4K WORDS OF NON-I/O
 9815                                                                            ;FOR 2 MEG WORDS OF MEMORY.
```

```
 9816  046402  012767  077406  123704          MOV     #77406,KIPDR6           ;SET KER PDR6 FOR 4KW R/W ACCESS.
 9817  046410  012737  000020  172516          MOV     #20,@#SR3               ;ENABLE 22 BIT ADDRESSING.
 9818  046416  012737  000001  177572          MOV     #1,@#SR0                ;TURN ON THE MMU.
 9819  046424  005737  157776                  TST     @#157776                ;ATTEMPT TO ADDRESS LAST MEMORY ADDR.
 9820                                  ;**************WILL TRAP TO 4 IF 2 MEG WORDS OF MEMORY NOT AVAILABLE*********
 9821  046430  013737  002762  000004          MOV     @#SLOC00,@#4            ;RESTORE LOC 4
 9822  046436  013737  002764  000006          MOV     @#SLOC01,@#6            ;RESTORE LOC 6
 9823  046444  005037  177572                  CLR     @#177572                ;MMU OFF
 9824  046450  005037  002776                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
 9825  046454  004767  063664                  JSR     PC,MMU                  ;INIT MMU
 9826  046460  012737  010000  177776          MOV     #10000,@#177776         ;INIT PSW
 9827  046466  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
 9828  046474  052737  001000  177746          BIS     #1000,@#177746          ;TURN CACHE TEST FEATURE ON
 9829  046502  012704  047240                  MOV     #PARVA3,R4              ;SET POINTERS TO INIT TABLES
 9830  046506  012701  047272                  MOV     #VIR3,R1                ;
 9831  046512  012437  172246          1$:     MOV     (R4)+,@#172246          ;INIT SIPAR3
 9832  046516  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9833  046524  012746  125252                  MOV     #125252,-(SP)           ;PUSH BACKGROUND DATA ON TO THE STACK
 9834  046530  006671  000000                  MTPI    @0(R1)                  ;WRITE DATA TO PHYSICAL ADDRESS
 9835  046534  006531                          MFPI    @(R1)+                  ;WRITE DATA AT PHYSICAL ADDRESS TO STACK
 9836  046536  022726  125252                  CMP     #125252,(SP)+           ;IS DATA EQUAL TO EXPECTED
 9837  046542  001403                          BEQ     2$                      ;YES GO ON
 9838  046544  005037  177572                  CLR     @#177572                ;TURN MMU OFF
 9839  046550  104002                          ERROR   +2                      ;MMU ERROR
 9840                                                                          ;NOT EQUAL GO TO ERROR
 9841  046552  005037  177572          2$:     CLR     @#177572                ;TURN MMU OFF
 9842  046556  021427  000333                  CMP     (R4),#333               ;ARE WE DONE
 9843  046562  001353                          BNE     1$                      ;NO GO TO 1$
 9844  046564  012704  047122                  MOV     #PARVA2,R4              ;SET POINTERS TO PAR INIT TABLES
 9845  046570  012701  047070                  MOV     #PARAD2,R1              ;
 9846  046574  012431                  3$:     MOV     (R4)+,@(R1)+            ;INIT PARS
 9847  046576  021127  000333                  CMP     (R1),#333               ;ARE WE DONE
 9848  046602  001374                          BNE     3$                      ;NO, GO TO 3$
 9849  046604  012704  047206                  MOV     #MODE2,R4               ;SET POINTERS TO ADDER PART B TABLES
 9850  046610  012701  047154                  MOV     #VIR2,R1                ;
 9851  046614  012702  047240                  MOV     #PARVA3,R2              ;
 9852  046620  012703  047272                  MOV     #VIR3,R3                ;
 9853  046624  004767  000076          4$:     JSR     PC,TS14                 ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
 9854                                                                          ;CHECK IF DATA AT PHYSICAL ADDRESS IS
 9855                                                                          ;EQUAL TO EXPECTED AND IF NOT DETERMINE
 9856                                                                          ;IF IT IS AN ADDER ERROR OR A MEMORY ERROR
 9857  046630  021127  000111                  CMP     (R1),#111               ;HAVE WE DONE ALL THE 22 BIT MODE I SPACE
 9858                                                                          ;CASES
 9859  046634  001373                          BNE     4$                      ;NO GO TO 4$
 9860  046636  005201                          INC     R1                      ;POINT TO 22 BIT MODE D SPACE CASE
 9861  046640  005201                          INC     R1                      ;
 9862  046642  005204                          INC     R4                      ;
 9863  046644  005204                          INC     R4                      ;
 9864  046646  012737  000027  172516          MOV     #27,@#172516            ;INIT MMR3
 9865  046654  012437  177776                  MOV     (R4)+,@#177776          ;INIT PSW
 9866  046660  012746  052525                  MOV     #52525,-(SP)            ;PUSH DATA ONTO STACK
 9867  046664  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9868  046672  106631                          MTPD    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS
 9869  046674  005037  177572                  CLR     @#177572                ;TURN MMU OFF
 9870  046700  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
 9871  046706  004767  000040                  JSR     PC,T14                  ;CHECK IF DATA AT PHYSICAL ADDRESS IS EQUAL
```

```
 9872
 9873
 9874  046712  005037  172516                CLR     @#172516                ;TO EXPECTED AND IF NOT DETERMINE IF IT
 9875  046716  004767  000004                JSR     PC,TS14                 ;IS AN ADDER ERROR OR A MEMORY ERROR
 9876                                                                          ;INIT MMR3 FOR 18 BIT MODE
 9877                                                                          ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
 9878                                                                          ;CHECK IF DATA AT PHYSICAL ADDRESS IS
 9879                                                                          ;EQUAL TO EXPECTED AND IF NOT DETERMINE IF
 9880  046722  000167  000376                JMP     T14FIN                  ;IT IS AN ADDER ERROR OR A MEMORY ERROR
 9881                                    ;
 9882                                    ;ROUTINE TO WRITE DATA TO PHYSICAL ADDRESS AND TO CHECK IF DATA AT
 9883                                    ;PHYSICAL ADDRESS IS EQUAL TO EXPECTED AND IF NOT DETERMINE IF IT IS
 9884                                    ;AN ADDER ERROR OR A MEMORY ERROR
 9885                                    ;
 9886  046726  012437  177776    TS14:    MOV     (R4)+,@#177776          ;INIT PSW
 9887  046732  012737  000001  177572     MOV     #1,@#177572             ;TURN MMU ON
 9888  046740  012746  052525             MOV     #52525,-(SP)            ;WRITE DATA ONTO STACK
 9889  046744  006631                     MTPI    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS VIA STACK
 9890  046746  005037  177572             CLR     @#177572                ;TURN MMU OFF
 9891  046752  012737  010000  177776 T14:  MOV     #10000,@#177776         ;INIT PSW
 9892  046760  012237  172246             MOV     (R2)+,@#172246          ;INIT SIPAR3
 9893  046764  012737  000001  177572     MOV     #1,@#177572             ;TURN MMU ON
 9894  046772  006573  000000             MFPI    @0(R3)                  ;DO RELOCATION
 9895  046776  022726  052525             CMP     #52525,(SP)+            ;IS DATA EQUAL TO EXPECTED
 9896  047002  001410                     BEQ     2$                      ;YES GO ON
 9897  047004  006573  000000             MFPI    @0(R3)                  ;WHAT TYPE OF ERROR IS IT
 9898  047010  022726  125252             CMP     #125252,(SP)+           ;
 9899  047014  001402                     BEQ     1$                      ;
 9900  047016  104002                     ERROR   .+2             ;MMU ERROR
 9901                                                                      ;IT IS A MEMORY ERROR
 9902  047020  000401                     BR      2$                      ;
 9903  047022  104002             1$:     ERROR   .+2             ;MMU ERROR
 9904                                                                      ;IT IS AN ADDER ERROR
 9905  047024  005037  177572     2$:     CLR     @#177572                ;TURN MMU OFF
 9906  047030  005203                     INC     R3                      ;
 9907  047032  005203                     INC     R3                      ;
 9908  047034  000207                     RTS     PC                      ;RETURN
 9909                                    ;
 9910                                    ;NON-EXISTANT MEMORY TRAP ROUTINE
 9911                                    ;
 9912  047036  005037  177572    NXMTRP: CLR     @#SR0                   ;TURN OFF MMU.
 9913  047042  012716  047324             MOV     #T14FIN,(SP)            ;SET UP STACK WITH RETURN ADDR.
 9914  047046  013737  002762  000004     MOV     @#SLOC00,@#4            ;RESTORE LOC 4
 9915  047054  013737  002764  000006     MOV     @#SLOC01,@#6            ;RESTORE LOC 6
 9916  047062  005037  177766             CLR     @#177766                ;CLEAR TIME OUT INDICATION FROM
 9917                                                                      ;CPU ERROR REGISTER.
 9918  047066  000006                     RTT                             ;RETURN FROM TRAP; GO TO NEXT TEST.
 9919                                    ;
 9920                                    ;ADDER TEST PART B TABLES
 9921                                    ;
 9922  047070  177646    PARAD2: .WORD   177646
 9923  047072  177650            .WORD   177650
 9924  047074  177652            .WORD   177652
 9925  047076  172240            .WORD   172240
 9926  047100  177640            .WORD   177640
 9927  047102  177642            .WORD   177642
```

```
 9928  047104  172244                          .WORD   172244
 9929  047106  177644                          .WORD   177644
 9930  047110  172252                          .WORD   172252
 9931  047112  172352                          .WORD   172352
 9932  047114  177662                          .WORD   177662
 9933  047116  172242                          .WORD   172242
 9934  047120  000333                          .WORD   333
 9935  047122  157700                PARVA2:   .WORD   157700
 9936  047124  137700                          .WORD   137700
 9937  047126  077700                          .WORD   077700
 9938  047130  176777                          .WORD   176777
 9939  047132  007600                          .WORD   007600
 9940  047134  167700                          .WORD   167700
 9941  047136  175700                          .WORD   175700
 9942  047140  177425                          .WORD   177425
 9943  047142  177220                          .WORD   177220
 9944  047144  173700                          .WORD   173700
 9945  047146  176700                          .WORD   176700
 9946  047150  077400                          .WORD   077400
 9947  047152  000333                          .WORD   333
 9948  047154  070000                VIR2:     .WORD   070000
 9949  047156  110000                          .WORD   110000
 9950  047160  130000                          .WORD   130000
 9951  047162  000000                          .WORD   000000
 9952  047164  000000                          .WORD   000000
 9953  047166  030000                          .WORD   030000
 9954  047170  050000                          .WORD   050000
 9955  047172  052524                          .WORD   052524
 9956  047174  136000                          .WORD   136000
 9957  047176  130000                          .WORD   130000
 9958  047200  000111                          .WORD   111
 9959  047202  030000                          .WORD   030000
 9960  047204  030000                          .WORD   030000
 9961  047206  030000                MODE2:    .WORD   030000
 9962  047210  030000                          .WORD   030000
 9963  047212  030000                          .WORD   030000
 9964  047214  010000                          .WORD   010000
 9965  047216  030000                          .WORD   030000
 9966  047220  030000                          .WORD   030000
 9967  047222  010000                          .WORD   010000
 9968  047224  030000                          .WORD   030000
 9969  047226  010000                          .WORD   010000
 9970  047230  000000                          .WORD   000000
 9971  047232  000111                          .WORD   111
 9972  047234  030000                          .WORD   030000
 9973  047236  010000                          .WORD   010000
 9974  047240  160000                PARVA3:   .WORD   160000
 9975  047242  140000                          .WORD   140000
 9976  047244  100000                          .WORD   100000
 9977  047246  176770                          .WORD   176770
 9978  047250  007600                          .WORD   007600
 9979  047252  170000                          .WORD   170000
 9980  047254  176000                          .WORD   176000
 9981  047256  177552                          .WORD   177552
 9982  047260  177400                          .WORD   177400
 9983  047262  174000                          .WORD   174000
```

```
 9984  047264  177000                          .WORD   177000
 9985  047266  007500                          .WORD   007500
 9986  047270  000333                          .WORD   333
 9987  047272  060000                  VIR3:   .WORD   060000
 9988  047274  060000                          .WORD   060000
 9989  047276  060000                          .WORD   060000
 9990  047300  060700                          .WORD   060700
 9991  047302  060000                          .WORD   060000
 9992  047304  060000                          .WORD   060000
 9993  047306  060000                          .WORD   060000
 9994  047310  060024                          .WORD   060024
 9995  047312  060000                          .WORD   060000
 9996  047314  060000                          .WORD   060000
 9997  047316  060000                          .WORD   060000
 9998  047320  060000                          .WORD   060000
 9999  047322  000333                          .WORD   333
10000
10001
10002                                  ;
10003  047324                          T14FIN:
10004
10005                                  ;       TEST NON-EXISTANT MEMORY TRAP
10006                                  ;;*****************************************************************
10007                                  ;WE ARE ASSUMING THAT THE NON-EXISTANT MEMORY TIME OUT
10008                                  ;FEATURE IS WORKING SINCE WE CAN'T GUARANTEE THAT
10009                                  ;THE SYSTEM BEING TESTED HAS A NON-EXISTANT MEMORY LOCATION.
10010                                  ;AT THIS TIME WE WILL ATTEMPT TO TEST THE NXM FUNCTION
10011  047324  004767  063014                  JSR     PC,MMU                  ;INIT THE MMU
10012  047330  012737  177400  172354          MOV     #177400,@#KIPAR6        ;SET KIPAR6 TO RELOCATE TO HIGHEST MEMORY
10013  047336  016767  130442  133416          MOV     4,SLOC00                ;SAVE VECTOR
10014  047344  016767  000026  130432          MOV     2$,4                    ;LOAD VEC WITH ADDR OF TRAP HANDLER
10015  047352  052767  000001  130212          BIS     #BIT00,SR0              ;TURN ON THE MMU
10016  047360  005067  130402                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10017  047364  005067  130406                  CLR     PS                      ;CLEAR THE PSW
10018  047370  005737  157776                  TST     @#157776                ;ACCESS PHYSICAL ADDR 17757776
10019  047374  000415                  1$:     BR      NXMFIN                  ;IF IT DOESN'T TRAP WE'LL ASSUME
10020                                                                          ;THAT THIS IA A 4 MEGABYTE SYSTEM
10021                                                                          ;AND GO TO THE NEXT TEST
10022  047376  022767  000040  130362  2$:     CMP     #BIT05,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
10023  047404  001401                          BEQ     3$                      ;
10024  047406  104002                          ERROR   .+2             ;MMU ERROR
10025  047410  022726  047374          3$:     CMP     #1$,(SP)+               ;IS CONTENTS OF STACK CORRECT?
10026  047414  001401                          BEQ     4$                      ;
10027  047416  104002                          ERROR   .+2             ;MMU ERROR
10028  047420  022726  000000          4$:     CMP     #0,(SP)+                ;IS CONTENTS OF STACK CORRECT?
10029  047424  001401                          BEQ     NXMFIN                  ;
10030  047426  104002                          ERROR   .+2             ;MMU ERROR
10031  047430  005067  130136          NXMFIN: CLR     SR0                     ;TURN OFF THE MMU
10032  047434  005067  130326                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10033  047440  016767  133316  130336          MOV     SLOC00,4                ;RESTORE THE VECTOR
10034
10035
10036  047446                          TSMM15:
10037                                  ;       PAGE WRITTEN BIT TEST
10038  047446  005037  177572                  CLR     @#177572                ;MMU OFF
10039  047452  005067  133320                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
```

```
10040  047456  004767  062662                  JSR     PC,MMU          ;INIT MMU
10041  047462  005037  177776                  CLR     @#177776        ;INIT PSW
10042  047466  012704  172300                  MOV     #172300,R4      ;SET POINTER TO KPDRS
10043  047472  004767  000114                  JSR     PC,T515         ;DO RELOCATIONS AND TEST KIPDRS FOR
10044                                                                  ;PAGE WRITTEN BIT BEING SET AND IF
10045                                                                  ;NOT SET GO TO ERROR
10046  047476  004767  000160                  JSR     PC,T15          ;DO RELOCATIONS AND TEST KDPDRS FOR
10047                                                                  ;PAGE WRITTEN BIT BEING SET AND IF NOT
10048                                                                  ;SET GO TO ERROR
10049  047502  012737  050000  177776          MOV     #50000,@#177776 ;INIT PSW
10050  047510  012704  172200                  MOV     #172200,R4      ;SET POINTER TO SPDRS
10051  047514  004767  000072                  JSR     PC,T515         ;DO RELOCATIONS AND TEST SIPDRS FOR
10052                                                                  ;PAGE WRITTEN BIT BEING SET AND IF NOT
10053                                                                  ;SET GO TO ERROR
10054  047520  004767  000136                  JSR     PC,T15          ;DO RELOCATIONS AND TEST SDPDRS FOR
10055                                                                  ;PAGE WRITTEN BIT BEING SET AND IF NOT
10056                                                                  ;SET GO TO ERROR
10057  047524  005037  177776                  CLR     @#177776        ;INIT PSW TO A KNOWN STATE
10058  047530  012737  170000  177776          MOV     #170000,@#177776 ;INIT PSW
10059  047536  012704  177600                  MOV     #177600,R4      ;SET POINTER TO UPDRS
10060  047542  004767  000044                  JSR     PC,T515         ;DO RELOCATIONS AND TEST UIPDRS FOR
10061                                                                  ;PAGE WRITTEN BIT BEING SET AND IF
10062                                                                  ;NOT SET GO TO ERROR
10063  047546  004767  000110                  JSR     PC,T15          ;DO RELOCATIONS AND TEST UDPDRS FOR
10064                                                                  ;PAGE WRITTEN BIT BEING SET AND IF NOT
10065                                                                  ;SET GO TO ERROR
10066  047552  005037  177776                  CLR     @#177776        ;INIT PSW TO A KNOWN STATE
10067  047556  012704  172300                  MOV     #172300,R4      ;SET POINTER TO KPDRS
10068  047562  004767  000152                  JSR     PC,T15A         ;EXPLICITLY WRITE TO KPDRS AND TEST
10069                                                                  ;FOR PAGE WRITTEN BIT BEING CLEARED
10070                                                                  ;AND IF NOT CLEARED GO TO ERROR
10071  047566  012704  172200                  MOV     #172200,R4      ;SET POINTER TO SPDRS
10072  047572  004767  000142                  JSR     PC,T15A         ;EXPLICITLY WRITE TO SPDRS AND TEST
10073                                                                  ;FOR PAGE WRITTEN BIT BEING CLEARED
10074                                                                  ;AND IF NOT CLEARED GO TO ERROR
10075  047576  012704  177600                  MOV     #177600,R4      ;SET POINTER TO UPDRS
10076  047602  004767  000132                  JSR     PC,T15A         ;EXPLICITLY WRITE TO UPDRS AND TEST
10077                                                                  ;FOR PAGE WRITTEN BIT BEING CLEARED
10078                                                                  ;AND IF NOT CLEARED GO TO ERROR
10079
10080  047606  000167  000154                  JMP     T15FIN
10081
10082                                  ;
10083                                  ;ROUTINE TO DO RELOCATIONS AND TEST IPDRS FOR PAGE WRITTEN BIT BEING
10084                                  ;SET AND IF NOT SET REPORT AN ERROR
10085                                  ;
10086  047612  005001                  T515:   CLR     R1              ;SET POINTER TO VIRTUAL ADDRESS
10087  047614  012737  000020  172516          MOV     #20,@#172516    ;INIT MMR3
10088  047622  012737  000001  177572  1$:     MOV     #1,@#177572     ;TURN MMU ON
10089  047630  011111                          MOV     (R1),(R1)       ;DO A RELOCATION
10090  047632  005037  177572                  CLR     @#177572        ;TURN MMU OFF
10091  047636  022427  077506                  CMP     (R4)+,#77506    ;IS DATA EQUAL TO EXPECTED
10092  047642  001401                          BEQ     2$              ;OK GO ON
10093  047644  104002                          ERROR   .2              ;MMU ERROR
                                                                       ;NO GO TO ERROR
10094  047646  062701  020000          2$:     ADD     #20000,R1       ;POINT TO NEXT VIRTUAL ADDRESS
10095  047652  020127  160000                  CMP     R1,#160000      ;ARE WE DONE
```

```
10096  047656  001361                        BNE     1$                      ;NO GO TO 1$
10097  047660  000207                        RTS     PC                      ;RETURN
10098                                ;
10099                                ;ROUTINE TO DO RELOCATIONS AND TEST DPDRS FOR PAGE WRITTEN BIT BEING SET
10100                                ;AND IF NOT SET REPORT AN ERROR
10101                                ;
10102  047662  005001                T15:    CLR     R1                      ;SET POINTER TO VIRTUAL ADDRESS
10103  047664  062704  000002              ADD     #2,R4                   ;POINT TO FIRST DPDR
10104  047670  012737  000027  172516      MOV     #27,@#172516            ;INIT MMR3
10105  047676  012737  000001  177572  1$: MOV     #1,@#177572             ;TURN MMU ON
10106  047704  011146                      MOV     (R1),-(SP)              ;PUSH DATA ONTO THE STACK
10107  047706  106611                      MTPD    (R_)                    ;DO A RELOCATION
10108  047710  005037  177572              CLR     @#177572                ;TURN MMU OFF
10109  047714  022427  077506              CMP     (R4)+,#077506           ;IS DATA EQUAL TO EXPECTED
10110  047720  001401                      BEQ     2$                      ;OK GO ON
10111  047722  104002                      ERROR   .2                  ;MMU ERROR
10112                                                                   ;NO GO TO ERROR
10113  047724  062701  020000          2$: ADD     #20000,R1               ;POINT TO NEXT VIRTUAL ADDRESS
10114  047730  020127  160000              CMP     R1,#160000              ;ARE WE DONE
10115  047734  001360                      BNE     1$                      ;NO GO TO 1$
10116  047736  000207                      RTS     PC                      ;RETURN
10117                                ;
10118                                ;ROUTINE TO EXPLICITLY WRITE TO PDRS AND TEST PAGE WRITTEN BIT FOR BEING
10119                                ;CLEARED AND IF NOT CLEARED REPORT AN ERROR
10120                                ;
10121  047740  005002                T15A:   CLR     R2                      ;CLEAR COUNTER
10122  047742  011414                1$:     MOV     (R4),(R4)               ;DO AN EXPLICIT WRITE TO PDR
10123  047744  022427  077406              CMP     (R4)+,#077406           ;IS DATA EQUAL TO EXPECTED
10124  047750  001401                      BEQ     2$                      ;OK GO ON
10125  047752  104002                      ERROR   .2                  ;MMU ERROR
10126                                                                   ;NO GO TO ERROR
10127  047754  005202                2$:     INC     R2                      ;INCREMENT POINTER
10128  047756  020227  000020              CMP     R2,#20                  ;ARE WE DONE
10129  047762  001367                      BNE     1$                      ;NO GO TO 1$
10130  047764  000207                      RTS     PC                      ;RETURN
10131  047766                        T15FIN:
10132                                ;
10133  047766                        TSMM16:
10134                                ;       TEST CSM (CALL SUPERVISOR MODE)
10135  047766  005037  177572              CLR     @#177572                ;MMU OFF
10136  047772  005037  002776              CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
10137  047776  012704  050316              MOV     #TMM16E,R4              ;INIT R4
10138  050002  004767  062336              JSR     PC,MMU                  ;INIT MMU
10139  050006  012737  000037  172516      MOV     #37,@#172516            ;ENABLE CSM INSTRUCTION
10140  050014  005037  177776              CLR     @#177776                ;SET PS TO KER MODE
10141  050020  013746  000010              MOV     @#10,-(SP)              ;SAVE VECTORS
10142  050024  013746  000014              MOV     @#14,-(SP)              ;
10143  050030  013746  000016              MOV     @#16,-(SP)              ;
10144  050034  012737  050206  000010      MOV     #TMM16B,@#10            ;SETUP NEW VECTORS
10145  050042  012737  000137  000014      MOV     #137,@#14               ;
10146  050050  012737  050326  000016      MOV     #TMM16A,@#16            ;
10147  050056  007014                      .WORD   7014                    ; TEST INSTRUCTION
10148  050060  104002                      ERROR   .2                  ;MMU ERROR
10149                                                                   ;GO TO ERROR IF NOT TRAPPED
10150  050062  012737  050236  000010  TSM16A: MOV     #TMM16C,@#10            ;SETUP NEW VECTOR
10151  050070  012737  000027  172516      MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
```

```
10152  050076  012737  140000  177776          MOV     #140000,@#177776        ;SET PS TO USER MODE
10153  050104  007014                          .WORD   7014                    ; TEST INSTRUCTION
10154  050106  104002                          ERROR   .2                      ;MMU ERROR
10155                                                                          ;GO TO ERROR IF NOT TRAPPED
10156  050110  012737  050266  000010  TSM16B: MOV     #TMM16D,@#10            ;SETUP NEW VECTOR
10157  050116  012737  000027  172516          MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
10158  050124  005037  177776                  CLR     @#177776                ;SET PS TO KER MODE
10159  050130  007014                          .WORD   7014                    ; TEST INSTRUCTION
10160  050132  104002                          ERROR   .2                      ;MMU ERROR
10161                                                                          ;GO TO ERROR IF NOT TRAPPED
10162  050134  012737  000037  172516  TSM16C: MOV     #37,@#172516            ;ENABLE CSM INSTRUCTION
10163  050142  012737  040000  177776          MOV     #40000,@#177776         ;SET PS TO SUP MODE
10164  050150  012706  000700                  MOV     #700,R6                 ;INIT SUP SP
10165  050154  012737  140000  177776          MOV     #140000,@#177776        ;SET PS TO USER MODE
10166  050162  012706  000600                  MOV     #600,R6                 ;INIT USER SP
10167  050166  012737  000014  000010          MOV     #14,@#10                ;SETUP NEW VECTOR
10168  050174  000277                          SCC                             ;SET ALL CC BITS
10169  050176  007024                          .WORD   7024                    ; TEST INSTRUCTION
10170  050200  104002                  TSM16D: ERROR   .2                      ;MMU ERROR
10171                                                                          ;GO TO ERROR IF NOT TRAPPED
10172  050202  000167  000504                  JMP     TM16A
10173                                  ;
10174                                  ;
10175  050206  042737  007777  177776  TMM16B: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10176  050214  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10177  050222  001401                          BEQ     1$                      ;YES GO ON
10178  050224  104002                          ERROR   .2                      ;MMU ERROR
10179                                                                          ;NO GO TO ERROR
10180  050226  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10181  050230  005726                          TST     (SP)+                   ;
10182  050232  000167  177624                  JMP     TSM16A                  ;CONTINUE TESTING
10183  050236  042737  007777  177776  TMM16C: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10184  050244  022737  030000  177776          CMP     #30000,@#177776         ;IS PS CORRECT
10185  050252  001401                          BEQ     1$                      ;YES GO ON
10186  050254  104002                          ERROR   .2                      ;MMU ERROR
10187                                                                          ;NO GO TO ERROR
10188  050256  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10189  050260  005726                          TST     (SP)+                   ;
10190  050262  000167  177622                  JMP     TSM16B                  ;CONTINUE TESTING
10191  050266  042737  007777  177776  TMM16D: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10192  050274  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10193  050302  001401                          BEQ     1$                      ;YES GO ON
10194  050304  104002                          ERROR   .2                      ;MMU ERROR
10195                                                                          ;NO GO TO ERROR
10196  050306  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10197  050310  005726                          TST     (SP)+                   ;
10198  050312  000167  177616                  JMP     TSM16C                  ;CONTINUE TESTING
10199  050316  156430                  TMM16E: .WORD   156430                  ; TEST LOCATION
10200  050320  104002                  TMM16F: ERROR   .2                      ;MMU ERROR
10201                                                                          ;GO TO ERROR IF DIDN'T ABORT
10202  050322  000167  000364                  JMP     TM16A
10203  050326  022737  070017  177776  TMM16A: CMP     #70017,@#177776         ;IS PS CORRECT
10204  050334  001401                          BEQ     1$                      ;YES GO ON
10205  050336  104002                          ERROR   .2                      ;MMU ERROR
10206                                                                          ;NO GO TO ERROR
10207  050340  020627  000572          1$:     CMP     R6,#572                 ;IS SP CORRECT
```

```
10208  050344  001401                        BEQ     2$                      ;YES GO ON
10209  050346  104002                        ERROR   .2                  ;MMU ERROR
10210                                                                      ;NO GO TO ERROR
10211  050350  020427  050320          2$:   CMP     R4,@TMM16E+2            ;IS R4 CORRECT
10212  050354  001401                        BEQ     3$                      ;YES GO ON
10213  050356  104002                        ERROR   .2                  ;MMU ERROR
10214                                                                      ;NO GO TO ERROR
10215  050360  023727  050316  156430  3$:   CMP     @#TMM16E,#156430        ;IS TEST LOCATION OK
10216  050366  001401                        BEQ     4$                      ;YES GO ON
10217  050370  104002                        ERROR   .2                  ;MMU ERROR
10218                                                                      ;NO GO TO ERROR
10219  050372  022627  156430          4$:   CMP     (SP)+,#156430           ;IS STACK CORRECT
10220  050376  001401                        BEQ     5$                      ;YES GO ON
10221  050400  104002                        ERROR   .2                  ;MMU ERROR
10222                                                                      ;NO GO TO ERROR
10223  050402  022627  050200          5$:   CMP     (SP)+,#TSM16D           ;IS STACK CORRECT
10224  050406  001401                        BEQ     6$                      ;YES GO ON
10225  050410  104002                        ERROR   .2                  ;MMU ERROR
10226                                                                      ;NO GO TO ERROR
10227  050412  0226[illegible]  140000  6$:   CMP     (SP)+,#140000           ;IS STACK CORRECT
10228  050416  001401                        BEQ     7$                      ;YES GO ON
10229  050420  104002                        ERROR   .2                  ;MMU ERROR
10230                                                                      ;NO GO TO ERROR
10231  050422  012706  000700          7$:   MOV     #700,R6                 ;RESTORE SUP SP
10232  050426  012737  140000  177776        MOV     #140000,@#177776        ;SET PS TO USER MODE
10233  050434  020627  000600                CMP     R6,#600                 ;IS USER SP CORRECT
10234  050440  001401                        BEQ     8$                      ;YES GO ON
10235  050442  104002                        ERROR   .2                  ;MMU ERROR
10236                                                                      ;NO GO TO ERROR
10237  050444  012767  077400  121526  8$:   MOV     #77400,SIPDR0           ;SETUP SIPDR0 TO ABORT
10238  050452  012737  050320  000016        MOV     #TMM16F,@#16            ;SETUP VECTOR
10239  050460  012737  000001  002776        MOV     #1,@#FLAG               ;SETUP FLAG FOR AN ABORT
10240  050466  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
10241  050474  010701                        MOV     R7,R1                   ;SAVE OLD PC
10242  050476  007014                        .WORD   7014                    ; TEST INSTRUCTION
10243  050500  022737  000000  002776        CMP     #0,@#FLAG               ;DID AN ABORT OCCUR
10244  050506  001401                        BEQ     9$                      ;YES GO ON
10245  050510  104002                        ERROR   .2                  ;MMU ERROR
10246                                                                      ;NO GO TO ERROR
10247  050512  023701  003010          9$:   CMP     @#SAVMR2,R1             ;IS MMR2 CORRECT
10248  050516  001401                        BEQ     10$                     ;YES GO ON
10249  050520  104002                        ERROR   .2                  ;MMU ERROR
10250                                                                      ;NO GO TO ERROR
10251  050522  023727  003004  100041  10$:   CMP     @#SAVMR0,#100041        ;IS MMR0 CORRECT
10252  050530  001401                        BEQ     11$                     ;YES GO ON
10253  050532  104002                        ERROR   .2                  ;MMU ERROR
10254                                                                      ;NO GO TO ERROR
10255  050534  012737  000037  172516  11$:   MOV     #37,@#172516            ;ENABLE CSM
10256  050542  012737  040000  177776        MOV     #40000,@#177776         ;SET PSW TO SUP
10257  050550  012706  000700                MOV     #700,R6                 ;SETUP SUP SP
10258  050554  012737  140000  177776        MOV     #140000,@#177776        ;SET PSW TO USE
10259  050562  012706  000600                MOV     #600,R6                 ;SETUP USE SP
10260  050566  012737  000014  000010        MOV     #14,@#10                ;SETUP NEW VECTOR
10261  050574  012737  050616  000016        MOV     #TS16,@#16              ;SETUP NEW VECTOR
10262  050602  000277                        SCC                             ;SET ALL CC BITS
10263  050604  007027                        .WORD   7027                    ;TEST INSTRUCTION
```

```
10264  050606  045712                          .WORD   45712
10265  050610  104002                  TS16A:  ERROR   .2              ;MMU ERROR
10266                                           ;GO TO ERROR IF DIDN'T TRAP
10267  050612  000167  000074                  JMP     TM16A
10268  050616  022737  070017  177776  TS16:   CMP     #70017,@#177776         ;IS PSW CORRECT
10269  050624  001401                          BEQ     200$            ;YES GO ON
10270  050626  104002                          ERROR   .2              ;MMU ERROR
10271                                           ;NO GO TO ERROR
10272  050630  020627  000572          200$:   CMP     R6,#572                 ;IS SP CORRECT
10273  050634  001401                          BEQ     201$            ;YES GO ON
10274  050636  104002                          ERROR   .2              ;MMU ERROR
10275                                           ;NO GO TO ERROR
10276  050640  022627  045712          201$:   CMP     (SP)+,#45712            ;IS STACK CORRECT
10277  050644  001401                          BEQ     202$            ;YES GO ON
10278  050646  104002                          ERROR   .2              ;MMU ERROR
10279                                           ;NO GO TO ERROR
10280  050650  022627  050610          202$:   CMP     (SP)+,#TS16A            ;IS STACK CORRECT
10281  050654  001401                          BEQ     203$            ;YES GO ON
10282  050656  104002                          ERROR   .2              ;MMU ERROR
10283                                           ;NO GO TO ERROR
10284  050660  022627  140000          203$:   CMP     (SP)+,#140000           ;IS STACK CORRECT
10285  050664  001401                          BEQ     204$            ;YES GO ON
10286  050666  104002                          ERROR   .2              ;MMU ERROR
10287                                           ;NO GO TO ERROR
10288  050670  012706  000700          204$:   MOV     #700,R6                 ;RESTORE SUP SP
10289  050674  012737  140000  177776          MOV     #140000,@#177776        ;SET PSW TO USER MODE
10290  050702  020627  000600                  CMP     R6,#600                 ;IS USER SP CORRECT
10291  050706  001401                          BEQ     TM16A           ;YES GO ON
10292  050710  104002                          ERROR   .2              ;MMU ERROR
10293                                           ;NO GO TO ERROR
10294  050712  005037  177776  TM16A:  CLR     @#177776                ;SET PS TO KER MODE
10295  050716  012637  000016                  MOV     (SP)+,@#16              ;RESTORE VECTORS
10296  050722  012637  000014                  MOV     (SP)+,@#14              ;
10297  050726  012637  000010                  MOV     (SP)+,@#10              ;
10298
10299                                   ;*****************************************************************************
10300                                   .SBTTL  FLOATING POINT TESTS
10301                                   ;*****************************************************************************
10302                                   ;                       BEGIN FLOATING POINT TESTING
10303                                   ;*****************************************************************************
10304                                   ;*****************************************************************************
10305  050732                          MBT1:
10306
10307                                   ;       FPP REGISTER BIT TESTS
10308                                   ;;*************************************************************
10309                                   ;R5=FPP POINTER
10310                                   ;R1=TEMPORARY COUNTER
10311                                   ;R2=POINTER TO EXPECTED DATA
10312                                   ;R3=POINTER TO RECEIVED DATA
10313                                   ;R4=ODD/EVEN COUNTER
10314  050732  170011                          SETD
10315  050734  005005                  MBT2:   CLR     R5                      ;SETUP FPP ACC POINTER
10316  050736  012702  003040                  MOV     #BTEXP,R2               ;POINT TO TEST DATA
10317  050742  012703  003050                  MOV     #BTRES,R3               ;POINT TO RECEIVED DATA
10318  050746  170400                  MBT2A:  CLRD    AC0                     ;SETUP FPP REGISTER VALUES
10319  050750  174012                          STD     AC0,(R2)                ;CLEAR EXPECTED VALUE
```

```
10320  050752  005004                          CLR     R4
10321  050754  170400                  BTGO:   CLRD    AC0             ;SETUP FPP REGISTER VALUES
10322  050756  170401                          CLRD    AC1
10323  050760  170402                          CLRD    AC2
10324  050762  170403                          CLRD    AC3
10325  050764  170404                          CLRD    AC4
10326  050766  170405                          CLRD    AC5
10327
10328  050770  010501                          MOV     R5,R1           ;GET FPP AC NUMBER INTO R1
10329  050772  070127  000014                  MUL     #14,R1          ;ALLOW 10 LOCATIONS FOR OPERATION
10330  050776  062701  051004                  ADD     #MAC0,R1        ;SETUP JMP LOCATION
10331  051002  000111                          JMP     (R1)
10332  051004  172467  132030          MAC0:   LDD     BTEXP,AC0       ;LOAD TEST DATA INTO TEST REGISTER
10333  051010  174067  132034          MAC0A:  STD     AC0,BTRES       ;SAVE TEST RESULT
10334  051014  000167  000074                  JMP     MACE            ;GET OUT
10335  051020  172567  132014          MAC1:   LDD     BTEXP,AC1       ;LOAD TEST DATA INTO TEST REGISTER
10336  051024  174167  132020                  STD     AC1,BTRES       ;SAVE TEST RESULT
10337  051030  000167  000060                  JMP     MACE            ;GET OUT
10338  051034  172667  132000          MAC2:   LDD     BTEXP,AC2       ;LOAD TEST DATA INTO TEST REGISTER
10339  051040  174267  132004                  STD     AC2,BTRES       ;SAVE TEST RESULT
10340  051044  000167  000044                  JMP     MACE            ;GET OUT
10341  051050  172767  131764          MAC3:   LDD     BTEXP,AC3       ;LOAD TEST DATA INTO TEST REGISTER
10342  051054  174367  131770                  STD     AC3,BTRES       ;SAVE TEST RESULT
10343  051060  000167  000030                  JMP     MACE            ;GET OUT
10344  051064  172467  131750          MAC4:   LDD     BTEXP,AC0
10345  051070  174004                          STD     AC0,AC4         ;LOAD TEST DATA INTO TEST REGISTER
10346  051072  172404                          LDD     AC4,AC0         ;SAVE TEST RESULT
10347  051074  000167  177710                  JMP     MAC0A           ;GET OUT
10348  051100  172467  131734          MAC5:   LDD     BTEXP,AC0       ;LOAD TEST DATA INTO TEST XFER REGISTER
10349  051104  174005                          STD     AC0,AC5         ;LOAD TEST REGISTER
10350  051106  172405                          LDD     AC5,AC0         ;STORE RESULT INTO XFER FPP REGISTER
10351  051110  000167  177674                  JMP     MAC0A           ;GET OUT
10352  051114  026767  131720  131726  MACE:   CMP     BTEXP,BTRES
10353  051122  001014                          BNE     BTER                    ;BRANCH IF REGISTER ERROR
10354  051124  026767  131712  131720          CMP     BTEXP+2,BTRES+2
10355  051132  001010                          BNE     BTER
10356  051134  026767  131704  131712          CMP     BTEXP+4,BTRES+4
10357  051142  001004                          BNE     BTER
10358  051144  026767  131676  131704          CMP     BTEXP+6,BTRES+6
10359  051152  001401                          BEQ     MBT8                    ;GOOD RESULT
10360  051154  104003                  BTER:   ERROR   +3              ;FPP ERROR
10361                                                                   ;FPP AC LOADED INCORRECTLY
10362                                  ;NOW VERIFY THE OTHER REGISTERS REMAINED ZERO
10363  051156                          MBT8:
10364  051156  005001                          CLR     R1                      ;CLEAR TEMPORARY COUNTER
10365  051160  005705                          TST     R5                      ;SEE IF R0 UNDER TEST
10366  051162  001411                          BEQ     MBT8A                   ;BRANCH IF TEST ING R0
10367  051164  020527  000004                  CMP     R5,#4                   ;SEE IF TESTING FPP REGISTER >R4
10368  051170  100006                          BPL     MBT8A                   ;SKIP R0 TESTING
10369  051172  174067  131652                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10370  051176  004767  000216                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10371  051202  001401                          BEQ     MBT8A                   ;BRANCH IF EXPECTED RESULT
10372  051204  104003                          ERROR   +3              ;FPP ERROR
10373                                                                   ;BAD AC0
10374  051206  020527  000001          MBT8A:  CMP     R5,#1                   ;SEE IF R1 UNDER TEST
10375  051212  001406                          BEQ     MBT8B                   ;BRANCH IF R1 UNDER TEST
```

```
10376  051214  174167  131630                  STD     AC1,BTRES               ;SAVE AC TEST RESULT
10377  051220  004767  000174                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10378  051224  001401                          BEQ     MBT8B                   ;BRANCH IF GOOD
10379  051226  104003                          ERROR   .3                  ;FPP ERROR
10380                                                                       ;BAD AC1
10381  051230  020527  000002          MBT8B:  CMP     R5,#2                   ;SEE IF TESTING FPP REGISTER AC2
10382  051234  001406                          BEQ     MBT8C                   ;BRANCH IF R2 UNDER TEST
10383  051236  174267  131606                  STD     AC2,BTRES               ;SAVE AC TEST RESULT
10384  051242  004767  000152                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10385  051246  001401                          BEQ     MBT8C                   ;BRANCH IF GOOD
10386  051250  104003                          ERROR   .3                  ;FPP ERROR
10387                                                                       ;BAD AC2
10388  051252  020527  000003          MBT8C:  CMP     R5,#3                   ;SEE IF R3 UNDER TEST
10389  051256  001406                          BEQ     MBT8D                   ;BRANCH IF R3 UNDER TEST
10390  051260  174367  131564                  STD     AC3,BTRES               ;SAVE AC TEST RESULT
10391  051264  004767  000130                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10392  051270  001401                          BEQ     MBT8D                   ;BRANCH IF GOOD
10393  051272  104003                          ERROR   .3                  ;FPP ERROR
10394                                                                       ;BAD AC3
10395  051274  020527  000004          MBT8D:  CMP     R5,#4                   ;SEE IF R4 UNDER TEST
10396  051300  001407                          BEQ     MBT8E                   ;BRANCH IF R4 UNDER TEST
10397  051302  172404                          LDD     AC4,AC0                 ;MOVE REGISTER CONTENT
10398  051304  174067  131540                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10399  051310  004767  000104                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10400  051314  001401                          BEQ     MBT8E                   ;BRANCH IF GOOD
10401  051316  104003                          ERROR   .3                  ;FPP ERROR
10402                                                                       ;BAD AC4
10403  051320  020527  000005          MBT8E:  CMP     R5,#5                   ;SEE IF R0 UNDER TEST
10404  051324  001407                          BEQ     MBT8F                   ;BRANCH IF R0 UNDER TEST
10405  051326  172405                          LDD     AC5,AC0                 ;MOVE REGISTER CONTENTS
10406  051330  174067  131514                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10407  051334  004767  000060                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10408  051340  001401                          BEQ     MBT8F                   ;BRANCH IF GOOD
10409  051342  104003                          ERROR   .3                  ;FPP ERROR
10410                                                                       ;BAD AC5
10411  051344  005204          MBT8F:  INC     R4                              ;INCREMENT PATTERN COUNTER
10412  051346  000241                  CLC
10413  051350  042704  177776                BIC     #177776,R4              ; TEST FOR ODD /EVEN
10414  051354  001401                          BEQ     MBT8FG                  ;BRANCH IF EVEN
10415  051356  000261                  SEC                                     ;SET CARRY FOR TEST PATTERN SHIFT
10416  051360  006112          MBT8FG: ROL     (R2)                            ;ROTATE L.SW OF TEST PATTERN
10417  051362  006162  000002                ROL     2(R2)                   ;ROTATE 2 WORD OF TEST PATTERN
10418  051366  006162  000004                ROL     4(R2)                   ;ROTATE 3 WORD OF TEST PATTERN
10419  051372  006162  000006                ROL     6(R2)                   ;ROTATE 4 WORD OF TEST PATTERN
10420  051376  103402                          BCS     MBT8I                   ;JUMP IF THROUGH WITH TEST PATTERN
10421  051400  000167  177350                  JMP     BTGO                    ;CONTINUE WITH NEW TEST PATTERN
10422
10423  051404  005205          MBT8I:  INC     R5                      ;GO TO NEXT REGISTER TEST
10424  051406  020527  000006                CMP     R5,#6                   ;SEE IF THROUGH TESTING
10425  051412  100016                          BPL     MBTE                    ;JUMP IF THROUGH
10426  051414  000167  177326                  JMP     MBT2A                   ;CONTINUE TESTING WITH NEW PATTERN
10427                                  ;
10428                                  ;
10429  051420  005767  131424          BTTST:  TST     BTRES                   ;VERIFY CONTENTS AS ZERO
10430  051424  001010                          BNE     BTTSTE                  ;EXIT IF NOT ZERO
10431  051426  005767  131420                  TST     BTRES+2                 ;VERIFY CONTENTS AS ZERO
```

```
10432  051432  001005                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10433  051434  005767  131414                  TST     BTRES+4         ;VERIFY CONTENTS AS ZERO
10434  051440  001002                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10435  051442  005767  131410                  TST     BTRES+6         ;VERIFY CONTENTS AS ZERO
10436  051446  000207                  BTTSTE: RTS     R7              ;GO BACK TO CALLING ROUTINE
10437
10438
10439                                          ;
10440                                          ;
10441  051450                          MBTE:
10442
10443                                  ;
10444  051450                          MFACU:
10445
10446                                  ;       TEST UNIQUENESS OF FPP ACCUMULATORS
10447                                  ;;*************************************************************
10448                                  ;THIS TEST LOADS UNIQUE PATTERNS INTO EACH ACCUMULATOR SIMULTANEOUSLY.
10449                                  ;R2=POINTER TO EXPECTED DATA
10450                                  ;R3=POINTER TO RECEIVED DATA
10451                                          ;
10452                                          ;
10453                                          ;
10454  051450  170011                          SETD
10455  051452  005000                  MFA:    CLR     R0                      ;SETUP FPP ACC POINTER
10456  051454  005004                          CLR     R4
10457  051456  012702  003040                  MOV     #BTEXP,R2               ;POINT TO TEST DATA
10458  051462  012703  003050                  MOV     #BTRES,R3               ;POINT TO RECEIVED DATA
10459  051466  012767  000051  131344          MOV     #51,BTEXP               ;SETUP EXPECTED DATA
10460  051474  012767  000052  131340          MOV     #52,BTEXP+2             ;
10461  051502  012767  000053  131334          MOV     #53,BTEXP+4             ;
10462  051510  012767  000054  131330          MOV     #54,BTEXP+6             ;
10463  051516  172467  131316                  LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10464  051522  174005                          STD     AC0,AC5                 ;PUT DATA INTO TEST REGISTER
10465  051524  004567  000210                  JSR     R5,SUBT                 ;SUBTRACT TEN FROM EACH EXPECTED DATA
10466  051530  172467  131304                  LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10467  051534  174004                          STD     AC0,AC4                 ;MOVE DATA INTO TEST REGISTER
10468  051536  004567  000176                  JSR     R5,SUBT                 ;SUBTRACT 10 FROM TEST DATA WORDS
10469  051542  172767  131272                  LDD     BTEXP,AC3               ;STORE INTO TEST REGISTER
10470  051546  004567  000166                  JSR     R5,SUBT                 ;GET NEXT SET OF UNIQUE DATA WORDS
10471  051552  172667  131262                  LDD     BTEXP,AC2               ;STORE INTO TEST REGISTER
10472  051556  004567  000156                  JSR     R5,SUBT                 ;GET NEXT SET OF TEST DATAS
10473  051562  172567  131252                  LDD     BTEXP,AC1               ;LOAD TEST REGISTER
10474  051566  004567  000146                  JSR     R5,SUBT                 ;GET NEXT SET OF TEST WORDS
10475  051572  172467  131242                  LDD     BTEXP,AC0               ;LOAD FINAL TEST REGISTER
10476                                                                          ;ALL REGISTER CONTAIN UNIQUE TEST WORDS
10477  051576  174067  131246                  STD     AC0,BTRES               ;STORE AC0,RESULT
10478  051602  004567  000216                  JSR     R5,BFA                  ;CHECK RESULT
10479  051606  001401                          BEQ     BFAC1                   ;BRANCH IF GOOD
10480  051610  104003                          ERROR   +3              ;FPP ERROR
10481                                                                  ;BAD AC0
10482  051612  004567  000154          BFAC1:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10483  051616  174167  131226                  STD     AC1,BTRES               ;STORE AC1 RESULT
10484  051622  004567  000176                  JSR     R5,BFA                  ;CHECK RESULT
10485  051626  001401                          BEQ     BFAC2                   ;BRANCH IF GOOD
10486  051630  104003                          ERROR   +3              ;FPP ERROR
10487                                                                  ;BAD RESULT AC1
```

```
10488  051632  004567  000134          BFAC2:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10489  051636  174267  131206                  STD     AC2,BTRES               ;STORE AC2 RESULT
10490  051642  004567  000156                  JSR     R5,BFA                  ;CHECK RESULT
10491  051646  001401                          BEQ     BFAC3                   ;BRANCH IF GOOD
10492  051650  104003                          ERROR   .3                  ;FPP ERROR
10493                                                                       ;BAD AC2 RESLUT
10494  051652  004567  000114          BFAC3:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10495  051656  174367  131166                  STD     AC3,BTRES               ;SAVE TEST RESULT
10496  051662  004567  000136                  JSR     R5,BFA                  ;CHECK RESULT
10497  051666  001401                          BEQ     BFAC4                   ;BRANCH IF GOOD
10498  051670  104003                          ERROR   .3                  ;FPP ERROR
10499                                                                       ;BAD AC3 RESULT
10500  051672  004567  000074          BFAC4:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10501  051676  172704                          LDD     AC4,AC3                 ;SAVE TEMPORARY
10502  051700  174367  131144                  STD     AC3,BTRES               ;STORE AC4 RESULT
10503  051704  004567  000114                  JSR     R5,BFA                  ;CHECK RESULT
10504  051710  001401                          BEQ     BFAC5                   ;BRANCH IF GOOD
10505  051712  104003                          ERROR   .3                  ;FPP ERROR
10506                                                                       ;BAD AC4 RESULT
10507  051714  004567  000052          BFAC5:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10508  051720  172605                          LDD     AC5,AC2                 ;SAVE TEMPORARY COPY
10509  051722  174267  131122                  STD     AC2,BTRES               ;MOVE AC5 RESULT
10510  051726  004567  000072                  JSR     R5,BFA                  ;CHECK RESULT
10511  051732  001454                          BEQ     BFAE                    ;BRANCH IF GOOD
10512  051734  104003                          ERROR   .3                  ;FPP ERROR
10513                                                                       ;BAD AC5 RESULT
10514  051736  000452                          BR      BFAE                    ;EXIT MODULE
10515
10516  051740  162767  000010  131072  SUBT:   SUB     #10,BTEXP                       ;UPDATE EXPECTED CONTENTS
10517  051746  162767  000010  131066          SUB     #10,BTEXP+2                     ;UPDATE EXPECTED CONTENTS
10518  051754  162767  000010  131062          SUB     #10,BTEXP+4                     ;UPDATE EXPECTED CONTENTS
10519  051762  162767  000010  131056          SUB     #10,BTEXP+6                     ;UPDATE EXPECTED CONTENTS
10520  051770  000205                          RTS     R5                      ;
10521  051772  062767  000010  131040  ADDT:   ADD     #10,BTEXP                       ;UPDATE EXPECTED CONTENTS
10522  052000  062767  000010  131034          ADD     #10,BTEXP+2                     ;UPDATE EXPECTED CONTENTS
10523  052006  062767  000010  131030          ADD     #10,BTEXP+4                     ;UPDATE EXPECTED CONTENTS
10524  052014  062767  000010  131024          ADD     #10,BTEXP+6                     ;UPDATE EXPECTED CONTENTS
10525  052022  000205                          RTS     R5                      ;
10526
10527  052024  026767  131010  131016  BFA:    CMP     BTEXP,BTRES             ;VERIFY CONTENTS
10528  052032  001013                          BNE     BFB                     ;EXIT IF NOT ZERO
10529  052034  026767  131002  131010          CMP     BTEXP+2,BTRES+2         ;VERIFY CONTENTS
10530  052042  001007                          BNE     BFB                     ;EXIT IF NOT ZERO
10531  052044  026767  130774  131002          CMP     BTEXP+4,BTRES+4         ;VERIFY CONTENTS
10532  052052  001003                          BNE     BFB                     ;EXIT IF NOT ZERO
10533  052054  026767  130766  130774          CMP     BTEXP+6,BTRES+6         ;VERIFY CONTENTS
10534  052062  000205                  BFB:    RTS     R5                      ;GO BACK TO CALLING ROUTINE
10535
10536
10537  052064                          BFAE:
10538
10539
10540
10541  052064                          TSFP1:
10542                                  ;       TEST LDFPS AND STFPS MODE 0
10543  052064  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
```

```
10544  052070  012704  147757                  MOV     #147757,R4              ;SETUP DATA TO BE LOADED
10545  052074  004767  000032                  JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10546  052100  012704  105252                  MOV     #105252,R4              ;SETUP DATA TO BE LOADED
10547  052104  004767  000022                  JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10548  052110  012704  042505                  MOV     #42505,R4               ;SETUP DATA TO BE LOADED
10549  052114  004767  000012                  JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10550  052120  005004                          CLR     R4                      ;SETUP DATA TO BE LOADED
10551  052122  004767  000004                  JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10552
10553                                  ;
10554  052126  000167  000014                  JMP     FIN1
10555                                  ;
10556  052132  170104                  LOST:   LDFPS   R4                      ;LOAD FPS WITH DATA
10557  052134  170201                          STFPS   R1                      ;LOAD R1 WITH (FPS)
10558  052136  020401                          CMP     R4,R1                   ;DID THE INSTRUCTIONS WORK
10559  052140  001401                          BEQ     1$                      ;YES GO ON
10560  052142  104003                          ERROR   +3              ;FPP ERROR
10561                                                                  ;NO GO TO ERROR
10562  052144  000207                  1$:     RTS     PC                      ;RETURN
10563  052146  000240                  FIN1:   NOP
10564                                  ;
10565  052150                          TSFP2:
10566                                  ;       TEST CFCC
10567  052150  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10568  052154  012704  000017                  MOV     #17,R4                  ;SETUP DATA TO BE LOADED
10569  052160  004767  000032                  JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10570  052164  012704  000012                  MOV     #12,R4                  ;SETUP DATA TO BE LOADED
10571  052170  004767  000022                  JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10572  052174  012704  000005                  MOV     #5,R4                   ;SETUP DATA TO BE LOADED
10573  052200  004767  000012                  JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10574  052204  005004                          CLR     R4                      ;SETUP DATA TO BE LOADED
10575  052206  004767  000004                  JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10576
10577                                  ;
10578  052212  000167  000024                  JMP     FIN2
10579                                  ;
10580  052216  170104                  TSF2:   LDFPS   R4                      ;LOAD FPS
10581  052220  170000                          CFCC                            ;COPY CONDITION CODES TO PS
10582  052222  013701  177776                  MOV     @#177776,R1             ;SAVE PS TO R1
10583  052226  042701  177760                  BIC     #177760,R1              ;MASK OUT UNWANTED BITS
10584  052232  020401                          CMP     R4,R1                   ;WAS CONDITION CODE BITS TRANSFERRED
10585                                                                          ;CORRECTLY
10586  052234  001401                          BEQ     1$                      ;YES GO ON
10587  052236  104003                          ERROR   +3              ;FPP ERROR
10588                                                                  ;NO GO TO ERROR
10589  052240  000207                  1$:     RTS     PC                      ;RETURN
10590  052242                          FIN2:
10591                                  ;
10592  052242                          TSFP3:
10593                                  ;       TEST SETF, SETD, SETI, SETL
10594  052242  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10595  052246  012704  000200                  MOV     #200,R4                 ;SETUP DATA TO BE LOADED
10596  052252  170104                          LDFPS   R4                      ;LOAD FPS
10597  052254  170001                          SETF                            ;MAKE FD=0
10598  052256  170201                          STFPS   R1                      ;STORE FPS
10599  052260  020127  000000                  CMP     R1,#0                   ;IS FD=0
```

```
10600  052264  001401                        BEQ     1$                      ;YES GO ON
10601  052266  104003                        ERROR   .3              ;FPP ERROR
10602                                                                 ;NO GO TO ERROR
10603  052270  170011                1$:     SETD                            ;MAKE FD=1
10604  052272  170201                        STFPS   R1                      ;STORE FPS
10605  052274  020104                        CMP     R1,R4                   ;IS FD=1
10606  052276  001401                        BEQ     2$                      ;YES GO ON
10607  052300  104003                        ERROR   .3              ;FPP ERROR
10608                                                                 ;NO GO TO ERROR
10609  052302  012704  000100        2$:     MOV     #100,R4                 ;SETUP DATA TO BE LOADED
10610  052306  170104                        LDFPS   R4                      ;LOAD FPS
10611  052310  170002                        SETI                            ;MAKE FL=0
10612  052312  170201                        STFPS   R1                      ;STORE FPS
10613  052314  020127  000000                CMP     R1,#0                   ;IS FL=0
10614  052320  001401                        BEQ     3$                      ;YES GO ON
10615  052322  104003                        ERROR   .3              ;FPP ERROR
10616                                                                 ;NO GO TO ERROR
10617  052324  170012                3$:     SETL                            ;MAKE FL=1
10618  052326  170201                        STFPS   R1                      ;STORE FPS
10619  052330  020104                        CMP     R1,R4                   ;IS FL=1
10620  052332  001401                        BEQ     4$                      ;YES GO ON
10621  052334  104003                        ERROR   .3              ;FPP ERROR
10622                                                                 ;NO GO TO ERROR
10623  052336                        4$:
10624                                        ;
10625  052336                        TSFP4:
10626                                ;       TEST ILLEGAL OP CODES AND STST
10627  052336  005037  132646                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10628  052342  012705  170003                MOV     #170003,R5              ;INIT OP CODE
10629  052346  013746  000244                MOV     @#244,-(SP)             ;SAVE FP VECTOR
10630  052352  012737  052502  000244        MOV     #ILLOP1,@#244           ;SETUP NEW VECTOR
10631  052360  013746  000004                MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
10632  052364  012737  052552  000004        MOV     #TIMEOU,@#4             ;SETUP NEW VECTOR
10633  052372  013746  000010                MOV     @#10,-(SP)              ;SAVE ILLEGAL VECTOR
10634  052376  012737  052556  000010        MOV     #ILLOP2,@#10            ;SETUP NEW VECTOR
10635  052404  005003                D1:     CLR     R3                      ;
10636  052406  170103                        LDFPS   R3                      ;CLEAR FPS
10637  052410  005002                        CLR     R2                      ;
10638  052412  010537  052416                MOV     R5,@#D2                 ;SETUP THE ILLEGAL INST
10639  052416  000000                D2:     .WORD   0                       ;
10640  052420  170000                D3:     CFCC                            ;MEMORY WORDS TO BE USED WITH
10641  052422  005202                        INC     R2                      ;EXECUTION OF ILLEGAL OP CODE
10642  052424  005202                        INC     R2                      ;
10643  052426  170201                        STFPS   R1                      ;SAVE FPS
10644  052430  104003                        ERROR   .3              ;FPP ERROR
10645                                                                 ;GO TO ERROR
10646  052432  022705  170010        D4:     CMP     #170010,R5              ;COMPUTE NEXT OP CODE
10647  052436  001003                        BNE     D5                      ;
10648  052440  012705  170013                MOV     #170013,R5              ;
10649  052444  000757                        BR      D1                      ;
10650  052446  022705  170077        D5:     CMP     #170077,R5              ;
10651  052452  001001                        BNE     D6                      ;
10652  052454  000402                        BR      D7                      ;
10653  052456  005205                D6:     INC     R5                      ;
10654  052460  000751                        BR      D1                      ;
10655  052462  012637  000010        D7:     MOV     (SP)+,@#10              ;RESTORE VECTORS
```

```
10656  052466  012637  000004                  MOV     (SP)+,@#4               ;
10657  052472  012637  000244                  MOV     (SP)+,@#244             ;
10658
10659                                          ;
10660  052476  000167  000060                  JMP     FIN4
10661                                          ;
10662  052502  022716  052420          ILLOP1: CMP     #D3,(SP)                        ;DID TRAP OCCUR ON TEST INST
10663  052506  001401                          BEQ     1$                              ;YES GO ON
10664  052510  104003                          ERROR   .+3                     ;FPP ERROR
10665                                                                          ;NO GO TO ERROR
10666  052512  022626                  1$:     CMP     (SP)+,(SP)+                     ;CLEAN UP STACK
10667  052514  170201                          STFPS   R1                              ;STORE FPS
10668  052516  022701  100000                  CMP     #100000,R1                      ;IS FPS CORRECT
10669  052522  001401                          BEQ     2$                              ;YES GO ON
10670  052524  104003                          ERROR   .+3                     ;FPP ERROR
10671                                                                          ;NO GO TO ERROR
10672  052526  005004                  2$:     CLR     R4                              ;INT R4 TO A KNOWN STATE
10673  052530  170304                          STST    R4                              ;STORE FEC AT R4
10674                                                                                  ;IF THE DESTINATION MODE IS IMPROPERLY
10675                                                                                  ;DECODED AN ODD ADDRESS TRAP TO 4
10676                                                                                  ;SHOULD OCCUR
10677  052532  022704  000002                  CMP     #2,R4                           ;IS FEC CORRECT
10678  052536  001002                          BNE     3$                              ;NO GO TO ERROR
10679  052540  000167  177666                  JMP     D4                              ;YES GO ON
10680  052544  104003                  3$:     ERROR   .+3                     ;FPP ERROR
10681                                                                          ;GO TO ERROR
10682  052546  000167  177660                  JMP     D4                              ;THEN GO ON
10683                                          ;
10684  052552  104003                  TIMEOU: ERROR   .+3                     ;FPP ERROR
10685                                                                          ;ERROR BECAUSE OF TRAP TO 4
10686  052554  000006                          RTT                                     ;RETURN
10687                                          ;
10688  052556  104003                  ILLOP2: ERROR   .+3                     ;FPP ERROR
10689                                                                          ;ERROR BECAUSE OF TRAP TO 10
10690  052560  000006                          RTT                                     ;RETURN
10691  052562                          FIN4:
10692                                          ;
10693  052562                          TSFP5:
10694                                  ;       TEST FID (INTERRUPT DISABLE BIT)
10695  052562  005037  132646                  CLR     @#TRPFLG                        ;CLEAR TRAP F'.AG
10696  052566  013746  000244                  MOV     @#244,-(SP)                     ;SAVE FP VECTOR
10697  052572  012737  052646  000244          MOV     #ILL,@#244                      ;SETUP NEW VECTOR
10698  052600  012703  040000                  MOV     #40000,R3                       ;SETUP DATA TO BE LOADED
10699  052604  170103                          LDFPS   R3                              ;LOAD FPS, FID=1
10700  052606  170020                          .WORD   170020                          ;ILLEGAL FP INSTRUCTION
10701  052610  170000                          CFCC
10702  052612  170201                          STFPS   R1                              ;SEE IF ERROR WAS RECORDED IN FPS
10703  052614  022701  140000                  CMP     #140000,R1                      ;
10704  052620  001401                          BEQ     1$                              ;YES GO ON
10705  052622  104003                          ERROR   .+3                     ;FPP ERROR
10706                                                                          ;NO GO TO ERROR
10707  052624  170304                  1$:     STST    R4                              ;SEE IF FEC=2
10708  052626  022704  000002                  CMP     #2,R4                           ;
10709  052632  001401                          BEQ     2$                              ;YES GO ON
10710  052634  104003                          ERROR   .+3                     ;FPP ERROR
10711                                                                          ;NO GO TO ERROR
```

```
10712  052636  012637  000244          2$:     MOV     (SP)+,@#244             ;RESTORE VECTOR
10713
10714                                   ;
10715  052642  000167  000004                  JMP     FIN5
10716                                   ;
10717  052646  104003                  ILL:    ERROR   +3              ;FPP ERROR
10718                                                                   ;FID ERROR
10719  052650  000006                          RTT                             ;RETURN
10720  052652                          FIN5:
10721                                   ;
10722  052652                          TSFP6:
10723                                   ;       TEST LDD, STD FSRC AND FDST MODE 1
10724  052652  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10725  052656  005004                          CLR     R4                      ;SETUP TO LOAD DATA
10726  052660  170104                          LDFPS   R4                      ;CLEAR FPS
10727  052662  170011                          SETD                            ;SET FD TO 1
10728  052664  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10729  052670  012737  053022  000004          MOV     #TSF6,@#4               ;SETUP NEW VECTOR
10730  052676  012704  053012                  MOV     #TS6DAT,R4              ;SETUP POINTER TO DATA
10731  052702  172414                          LDD     (R4),AC0                ; TEST INSTRUCTION
10732  052704  020427  053012                  CMP     R4,#TS6DAT              ;IS R4 CORRECT
10733  052710  001401                          BEQ     1$                      ;YES GO ON
10734  052712  104003                          ERROR   +3              ;FPP ERROR
10735                                                                   ;NO GO TO ERROR
10736  052714  012701  053002          1$:     MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
10737  052720  012703  000004                  MOV     #4,R3                   ;INIT COUNTER
10738  052724  022421                  2$:     CMP     (R4)+,(R1)+             ;WAS SOURCE DATA ALTERED
10739  052726  001401                          BEQ     3$                      ;NO GO ON
10740  052730  104003                          ERROR   +3              ;FPP ERROR
10741                                                                   ;YES GO TO ERROR
10742  052732  077304                  3$:     SOB     R3,2$                   ;ARE WE DONE
10743  052734  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER FOR DATA
10744  052740  174014                          STD     AC0,(R4)                ; TEST INSTRUCTION
10745  052742  020427  003122                  CMP     R4,#TSTLOC                      ;IS R4 CORRECT
10746  052746  001401                          BEQ     4$                      ;YES GO ON
10747  052750  104003                          ERROR   +3              ;FPP ERROR
10748                                                                   ;NO GO TO ERROR
10749  052752  012701  053002          4$:     MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
10750  052756  012703  000004                  MOV     #4,R3                   ;INIT COUNTER
10751  052762  022421                  5$:     CMP     (R4)+,(R1)+             ;IS DESTINATION DATA CORRECT
10752  052764  001401                          BEQ     6$                      ;YES GO ON
10753  052766  104003                          ERROR   +3              ;FPP ERROR
10754                                                                   ;NO GO TO ERROR
10755  052770  077304                  6$:     SOB     R3,5$                   ;ARE WE DONE
10756  052772  012637  000004                  MOV     (SP)+,@#4               ;RESTORE VECTOR
10757
10758                                   ;
10759  052776  000167  000024                  JMP     FIN6
10760                                   ;
10761  053002  177777                  TS6DA:  .WORD   177777
10762  053004  000000                          .WORD   000000
10763  053006  052525                          .WORD   052525
10764  053010  125252                          .WORD   125252
10765  053012  177777                  TS6DAT: .WORD   177777
10766  053014  000000                          .WORD   000000
10767  053016  052525                          .WORD   052525
```

```
10768  053020  125252                          .WORD   125252
10769
10770  053022  104003                  ;TSF6:  ERROR   .3              ;FPP ERROR
10771                                                                  ;ODD ADDRESS TRAP
10772  053024  000006                          RTT                     ;RETURN
10773  053026                          FIN6:
10774                                  ;
10775  053026                          TSFP7:
10776                                  ;       TEST LDD, LDF FSRC MODE 0
10777  053026  005037  132646                  CLR     @#TRPFLG        ;CLEAR TRAP FLAG
10778  053032  012704  000200                  MOV     #200,R4         ;SETUP TO LOAD FPS
10779  053036  170104                          LDFPS   R4              ;LOAD FPS, FD=1
10780  053040  013746  000004                  MOV     @#4,-(SP)       ;SAVE TIMEOUT VECTOR
10781  053044  012737  053212  000004          MOV     #TSF7,@#4       ;SETUP NEW VECTOR
10782  053052  012704  053216                  MOV     #TS7D^1,R4      ;SETUP POINTER TO DATA
10783  053056  172414                          LDD     (R4),AC0        ;CLEAR AC0
10784  053060  012701  053226                  MOV     #TS7DA2,R1      ;SETUP POINTER TO DATA
10785  053064  172511                          LDD     (R1),AC1        ;LOAD AC1 WITH DATA
10786  053066  172401                          LDD     AC1,AC0         ; TEST INSTRUCTION
10787  053070  012704  003122                  MOV     #TSTLOC,R4      ;
10788  053074  174114                          STD     AC1,(R4)        ;CHECK IF AC1 HAS BEEN ALTERED
10789  053076  004767  000072                  JSR     PC,CHECK7       ;
10790  053102  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTERS FOR DATA
10791  053106  012701  053226                  MOV     #TS7DA2,R1      ;
10792  053112  174014                          STD     AC0,(R4)        ;CHECK IF AC0 RECEIVED CORRECT DATA
10793  053114  004767  000054                  JSR     PC,CHECK7       ;
10794  053120  012701  053216                  MOV     #TS7DA1,R1      ;SETUP POINTER TO DATA
10795  053124  172511                          LDD     (R1),AC1        ;CLEAR AC1
10796  053126  170001                          SETF                    ;SET FD=0
10797  053130  172401                          LDF     AC1,AC0         ; TEST INSTRUCTION
10798  053132  170011                          SETD                    ;SET FD=1
10799  053134  012704  003122                  MOV     #TSTLOC,R4      ;SETUP POINTER TO DATA
10800  053140  174114                          STD     AC1,(R4)        ;CHECK IF AC1 HAS BEEN ALTERED
10801  053142  004767  000026                  JSR     PC,CHECK7       ;
10802  053146  012704  053236                  MOV     #TS7DA4,R4      ;SETUP POINTERS FOR DATA
10803  053152  012701  003122                  MOV     #TSTLOC,R1      ;
10804  053156  174011                          STD     AC0,(R1)        ;CHECK IF AC0 HAS CORRECT DATA
10805  053160  004767  000010                  JSR     PC,CHECK7       ;
10806  053164  012637  000004                  MOV     (SP)+,@#4       ;RESTORE VECTOR
10807
10808                                  ;
10809  053170  000167  000052                  JMP     FIN7
10810                                  ;
10811  053174  012703  000004          CHECK7: MOV     #4,R3                   ;INIT COUNTER
10812  053200  022421                  CHEK7:  CMP     (R4)+,(R1)+             ;IS DATA OK
10813  053202  001401                          BEQ     CHK7                    ;YES GO ON
10814  053204  104003                          ERROR   .3              ;FPP ERROR
10815                                                                  ;NO GO TO ERROR
10816  053206  077304                  CHK7:   SOB     R3,CHEK7                ;ARE WE DONE
10817  053210  000207                          RTS     PC                      ;YES RETURN
10818
10819  053212  104003                  TSF7:   ERROR   .3              ;FPP ERROR
10820                                                                  ;ODD ADDRESS TRAP
10821  053214  000006                          RTT                     ;RETURN
10822
10823  053216  000000                  TS7DA1: .WORD   0
```

```
10824  053220  000000                          .WORD   0
10825  053222  000000                          .WORD   0
10826  053224  000000                          .WORD   0
10827  053226  037641                  TS7DA2: .WORD   37641
10828  053230  065121                          .WORD   65121
10829  053232  037373                          .WORD   37373
10830  053234  022265                          .WORD   22265
10831  053236  000000                  TS7DA4: .WORD   0
10832  053240  000000                          .WORD   0
10833  053242  037373                          .WORD   37373
10834  053244  022265                          .WORD   22265
10835  053246                          FIN7:
10836                                  ;
10837  053246                          TSFP10:
10838                                  ;       TEST STD, STF FDST MODE 0
10839  053246  005037  132646                 CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10840  053252  012704  000200                 MOV     #200,R4                 ;SETUP TO LOAD FPS
10841  053256  170104                          LDFPS   R4                      ;LOAD FPS, FD=1
10842  053260  013746  000004                 MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10843  053264  012737  053432  000004         MOV     #TSF10,@#4              ;SETUP NEW VECTOR
10844  053272  012704  053436                 MOV     #TS10D1,R4              ;SETUP POINTER TO DATA
10845  053276  172414                          LDD     (R4),AC0                ;CLEAR AC0
10846  053300  012701  053446                 MOV     #TS10D2,R1              ;SETUP POINTER TO DATA
10847  053304  172511                          LDD     (R1),AC1                ;LOAD AC1 WITH DATA
10848  053306  174100                          STD     AC1,AC0                 ; TEST INSTRUCTION
10849  053310  012704  003122                 MOV     #TSTLOC,R4              ;
10850  053314  174114                          STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
10851  053316  004767  000072                 JSR     PC,CHEC10               ;
10852  053322  012704  003122                 MOV     #TSTLOC,R4              ;SETUP POINTERS FOR DATA
10853  053326  012701  053446                 MOV     #TS10D2,R1              ;
10854  053332  174014                          STD     AC0,(R4)                ;CHECK IF AC0 RECEIVED CORRECT DATA
10855  053334  004767  000054                 JSR     PC,CHEC10               ;
10856  053340  012701  053436                 MOV     #TS10D1,R1              ;SETUP POINTER TO DATA
10857  053344  172511                          LDD     (R1),AC1                ;CLEAR AC1
10858  053346  170001                          SETF                            ;SET FD=0
10859  053350  174100                          STF     AC1,AC0                 ; TEST INSTRUCTION
10860  053352  170011                          SETD                            ;SET FD=1
10861  053354  012704  003122                 MOV     #TSTLOC,R4              ;SETUP POINTER TO DATA
10862  053360  174114                          STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
10863  053362  004767  000026                 JSR     PC,CHEC10               ;
10864  053366  012704  053456                 MOV     #TS10D4,R4              ;SETUP POINTERS FOR DATA
10865  053372  012701  003122                 MOV     #TSTLOC,R1              ;
10866  053376  174011                          STD     AC0,(R1)                ;CHECK IF AC0 HAS CORRECT DATA
10867  053400  004767  000010                 JSR     PC,CHEC10               ;
10868  053404  012637  000004                 MOV     (SP)+,@#4               ;RESTORE VECTOR
10869
10870                                  ;
10871  053410  000167  000052                 JMP     FIN10
10872                                  ;
10873  053414  012703  000004         CHEC10: MOV     #4,R3                   ;INIT COUNTER
10874  053420  022421                  CH10:   CMP     (R4)+,(R1)+             ;IS DATA OK
10875  053422  001401                          BEQ     CHK10                   ;YES GO ON
10876  053424  104003                          ERROR   +3              ;FPP ERROR
10877                                                                  ;NO GO TO ERROR
10878  053426  077304                  CHK10:  SOB     R3,CH10                 ;ARE WE DONE
10879  053430  000207                          RTS     PC                      ;YES RETURN
```

```
10880
10881  053432  104003                  ;
                                TSF10:  ERROR   .3              ;FPP ERROR
10882                                                           ;ODD ADDRESS TRAP
10883  053434  000006                          RTT                     ;
10884
                                ;
10835  053436  000000          TS10D1: .WORD   0
10886  053440  000000                  .WORD   0
10887  053442  000000                  .WORD   0
10888  053444  000000                  .WORD   0
10889  053446  177777          TS10D2: .WORD   177777
10890  053450  111236                  .WORD   111236
10891  053452  100045                  .WORD   100045
10892  053454  003651                  .WORD   3651
10893  053456  000000          TS10D4: .WORD   0
10894  053460  000000                  .WORD   0
10895  053462  100045                  .WORD   100045
10896  053464  003651                  .WORD   3651
10897  053466                  FIN10:
10898                           ;
10899  053466                  TSFP11:
10900                           ;       TEST FDST SINGLE OPERAND MODE 0
10901  053466  005037  132646                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10902  053472  012704  000200                MOV     #200,R4                 ;SETUP TO LOAD FPS
10903  053476  170104                        LDFPS   R4                      ;SET FD=1
10904  053500  012704  053552                MOV     #TS11D1,R4              ;SETUP POINTER TO DATA
10905  053504  172414                        LDD     (R4),AC0                ;LOAD ALL ONES TO AC0
10906  053506  170400                        CLRD    AC0                     ; TEST INSTRUCTION
10907  053510  170203                        STFPS   R3                      ;GET FPS
10908  053512  012704  003122                MOV     #TSTLOC,R4              ;
10909  053516  174014                        STD     AC0,(R4)                ;CHECK AC0 FOR ALL ZEROES
10910  053520  012701  000004                MOV     #4,R1                   ;INIT COUNTER
10911  053524  022427  000000        1$:     CMP     (R4)+,#0                ;
10912  053530  001401                        BEQ     2$                      ;OK GO ON
10913  053532  104003                        ERROR   .3              ;FPP ERROR
10914                                                           ;NO GO TO ERROR
10915  053534  077105                2$:     SOB     R1,1$                   ;ARE WE DONE
10916  053536  020327  000204                CMP     R3,#0204                ;CHECK FPS
10917  053542  001401                        BEQ     3$                      ;OK GO ON
10918  053544  104003                        ERROR   .3              ;FPP ERROR
10919                                                           ;NO GO TO ERROR
10920  053546                  3$:
10921                           ;
10922  053546  000167  000010                JMP     FIN11
10923                           ;
10924  053552  177777          TS11D1: .WORD   177777
10925  053554  177777                  .WORD   177777
10926  053556  177777                  .WORD   177777
10927  053560  177777                  .WORD   177777
10928  053562                  FIN11:
10929                           ;
10930  053562                  TSFP12:
10931                           ;       TEST FDST SOP MODE 0 WITH ILLEGAL AC7
10932  053562  005037  132646                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10933  053566  012703  040200                MOV     #40200,R3               ;SETUP TO LOAD FPS
10934  053572  170103                        LDFPS   R3                      ;SET FID=1, AND FD=1
10935  053574  170407                        CLRD    AC7                     ; TEST INSTRUCTION
```

```
10936  053576  170204                          STFPS   R4                      ;GET FPS
10937  053600  170305                          STST    R5                      ;GET FEC
10938  053602  022704  140200                  CMP     #140200,R4              ;IS FPS CORRECT
10939  053606  001401                          BEQ     1$                      ;YES GO ON
10940  053610  104003                          ERROR   .3              ;FPP ERROR
10941                                                                  ;NO GO TO ERROR
10942  053612  022705  000002          1$:     CMP     #2,R5                   ;IS FEC CORRECT
10943  053616  001401                          BEQ     2$                      ;YES GO ON
10944  053620  104003                          ERROR   .3              ;FPP ERROR
10945                                                                  ;NO GO TO ERROR
10946  053622                          2$:
10947                                  ;
10948  053622                          TSFP13:
10949                                  ;       TEST FDST SOP MODE 1
10950  053622  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10951  053626  012737  053742  000004          MOV     #TSF13,@#4              ;SETUP NEW VECTOR
10952  053634  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10953  053640  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
10954  053644  170102                          LDF     R2                      ;SET FD=1
10955  053646  012705  000004                  MOV     #4,R5                   ;INIT COUNTER
10956  053652  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
10957  053656  012724  177777          100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
10958  053662  077503                          SOB     R5,100$                 ;ARE WE DONE
10959  053664  012702  003122                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
10960  053670  170412                          CLRD    (R2)                    ; TEST INSTRUCTION
10961  053672  170203                          STFPS   R3                      ;GET FPS
10962  053674  020227  003122                  CMP     R2,#TSTLOC                      ;WAS R2 ALTERED
10963  053700  001401                          BEQ     1$                      ;NO GO ON
10964  053702  104003                          ERROR   .3              ;FPP ERROR
10965                                                                  ;YES GO TO ERROR
10966  053704  012701  000004          1$:     MOV     #4,R1                   ;INIT COUNTER
10967  053710  022227  000000          2$:     CMP     (R2)+,#0                ;CHECK LOCATION FOR 0
10968  053714  001401                          BEQ     3$                      ;OK GO ON
10969  053716  104003                          ERROR   .3              ;FPP ERROR
10970                                                                  ;NO GO TO ERROR
10971  053720  077105                  3$:     SOB     R1,2$                   ;ARE WE DONE
10972  053722  020327  000204                  CMP     R3,#204                 ;CHECK FPS
10973  053726  001401                          BEQ     4$                      ;OK GO ON
10974  053730  104003                          ERROR   .3              ;FPP ERROR
10975                                                                  ;NO GO TO ERROR
10976  053732  012637  000004          4$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
10977
10978                                  ;
10979  053736  000167  000004                  JMP     FIN13
10980                                  ;
10981  053742  104003                  TSF13:  ERROR   .3              ;FPP ERROR
10982                                                                  ;ODD ADDRESS TRAP
10983  053744  000006                          RTT                             ;RETURN
10984                                  ;
10985  053746                          FIN13:
10986                                  ;
10987  053746                          TSFP14:
10988                                  ;       TEST FDST SOP MODE 2
10989  053746  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10990  053752  012737  054072  000004          MOV     #TSF14,@#4              ;SETUP NEW VECTOR
10991  053760  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
```

```
10992  053764  012702  000200          MOV     #200,R2                 ;SETUP TO LOAD FPS
10993  053770  170102                  LDFPS   R2                      ;SET FD=1
10994  053772  012705  000004          MOV     #4,R5                   ;INIT COUNTER
10995  053776  012704  003122          MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
10996  054002  012724  177777  100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
10997  054006  077503                  SOB     R5,100$                 ;ARE WE DONE
10998  054010  012702  003122          MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
10999  054014  170422                  CLRD    (R2)+                   ; TEST INSTRUCTION
11000  054016  170203                  STFPS   R3                      ;GET FPS
11001  054020  020227  003132          CMP     R2,#TSTLOC+10                   ;IS R2 CORRECT
11002  054024  001401                  BEQ     1$                      ;YES GO ON
11003  054026  104003                  ERROR   +3              ;FPP ERROR
11004                                                           ;NO GO TO ERROR
11005  054030  012702  003122  1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11006  054034  012701  000004          MOV     #4,R1                   ;INIT COUNTER
11007  054040  022227  000000  2$:     CMP     (R2)+,#0                ;CHECK LOCATION FOR 0
11008  054044  001401                  BEQ     3$                      ;YES GO ON
11009  054046  104003                  ERROR   +3              ;FPP ERROR
11010                                                           ;NO GO TO ERROR
11011  054050  077105          3$:     SOB     R1,2$                   ;ARE WE DONE
11012  054052  020327  000204          CMP     R3,#204                 ;CHECK FPS
11013  054056  001401                  BEQ     4$                      ;OK GO ON
11014  054060  104003                  ERROR   +3              ;FPP ERROR
11015                                                           ;NO GO TO ERROR
11016  054062  012637  000004  4$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11017
11018                          ;
11019  054066  000167  000004          JMP     FIN14
11020                          ;
11021  054072  104003          TSF14:  ERROR   +3              ;FPP ERROR
11022                                                           ;000 ADDRESS TRAP
11023  054074  000006                  RTT                             ;RETURN
11024                          ;
11025  054076  000240          FIN14:  NOP
11026                          ;
11027                          ;
11028                          ;---------------------------------
11029                          ;TEST  FDST SOP MODE 3
11030                          ;
11031                          ;
11032  054100                  TSFP15:
11033                          ;
11034  054100  013746  000004          MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11035  054104  012737  054260  000004  MOV     #TSF15,@#4              ;SETUP NEW VECTOR
11036  054112  005037  132646          CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11037  054116  012702  000200          MOV     #200,R2                 ;SETUP TO LOAD FPS
11038  054122  170102                  LDFPS   R2                      ;SET FD=1
11039  054124  012705  000011          MOV     #9.,R5                  ;INIT COUNTER
11040  054130  012704  003122          MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11041  054134  012724  177777  100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11042  054140  077503                  SOB     R5,100$                 ;ARE WE DONE
11043  054142  012737  003134  003122  MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11044  054150  012702  003122          MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11045  054154  170432                  CLRD    @(R2)+                  ; TEST INSTRUCTION
11046  054156  170203                  STFPS   R3                      ;GET FPS
11047  054160  020227  003124          CMP     R2,#TSTLOC+2                    ;IS R2 CORRECT
```

```
11048  054164  001401                        BEQ     1$                      ;YES GO ON
11049  054166  104003                        ERROR   .3                  ;FPP ERROR
11050                                                                     ;NO GO TO ERROR
11051  054170  012702  003122        1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11052  054174  022227  003134                CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11053  054200  001401                        BEQ     2$                      ;YES GO ON
11054  054202  104003                        ERROR   .3                  ;FPP ERROR
11055                                                                     ;NO GO TO ERROR
11056  054204  012701  000004        2$:     MOV     #4,R1                   ;INIT COUNTER
11057  054210  022227  177777        3$:     CMP     (R2)+,#177777           ;IS LOCATION ALL ONES
11058  054214  001401                        BEQ     4$                      ;YES GO ON
11059  054216  104003                        ERROR   .3                  ;FPP ERROR
11060                                                                     ;NO GO TO ERROR
11061  054220  077105                4$:     SOB     R1,3$                   ;ARE WE DONE
11062  054222  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11063  054226  022227  000000        5$:     CMP     (R2)+,#0                ;IS LOCATION 0
11064  054232  001401                        BEQ     6$                      ;YES GO ON
11065  054234  104003                        ERROR   .3                  ;FPP ERROR
11066                                                                     ;NO GO TO ERROR
11067  054236  077105                6$:     SOB     R1,5$                   ;ARE WE DONE
11068  054240  020327  000204                CMP     R3,#204                 ;CHECK FPS
11069  054244  001401                        BEQ     7$                      ;OK GO ON
11070  054246  104003                        ERROR   .3                  ;FPP ERROR
11071                                                                     ;NO GO TO ERROR
11072  054250  012637  000004        7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11073
11074                                  ;
11075  054254  000167  000004                JMP     FIN15
11076                                  ;
11077  054260  104003                TSF15:  ERROR   .3                  ;FPP ERROR
11078                                                                     ;ODD ADDRESS TRAP
11079  054262  000006                        RTT                             ;RETURN
11080                                  ;
11081  054264  000240                FIN15:  NOP
11082                                  ;
11083                                  ;
11084                                  ;---------------------------------
11085                                  ;TEST  FDST SOP MODE 4
11086                                  ;
11087                                  ;
11088  054266                        TSFP16:
11089                                  ;
11090  054266  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11091  054272  012737  054424  000004        MOV     #TSF16,@#4              ;SETUP NEW VECTOR
11092  054300  005037  132646                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11093  054304  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11094  054310  170102                        LDFPS   R2                      ;SET FD=1
11095  054312  012705  000010                MOV     #8.,R5                  ;INIT COUNTER
11096  054316  012704  003122                MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11097  054322  012724  177777        100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11098  054326  077503                        SOB     R5,100$                 ;ARE WE DONE
11099  054330  012702  003132                MOV     #TSTLOC+10,R2                   ;SETUP POINTER TO DATA
11100  054334  170442                        CLRD    -(R2)                   ; TEST INSTRUCTION
11101  054336  170203                        STFPS   R3                      ;GET FPS
11102  054340  020227  003122                CMP     R2,#TSTLOC                      ;IS R2 CORRECT
11103  054344  001401                        BEQ     1$                      ;YES GO ON
```

```
11104  054346  104003                          ERROR   +3              ;FPP ERROR
11105                                                                  ;NO GO TO ERROR
11106  054350  012701  000004          1$:     MOV     #4,R1                   ;INIT COUNTER
11107  054354  022227  000000          2$:     CMP     (R2)+,#0                ;IS LOCATION C
11108  054360  001401                          BEQ     3$                      ;YES GO ON
11109  054362  104003                          ERROR   +3              ;FPP ERROR
11110                                                                  ;NO GO TO ERROR
11111  054364  077105                  3$:     SOB     R1,2$                   ;ARE WE DONE
11112  054366  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11113  054372  022227  177777          4$:     CMP     (R2)+,#177777           ;IS LOCATION UNCHANGED
11114  054376  001401                          BEQ     5$                      ;YES GO ON
11115  054400  104003                          ERROR   +3              ;FPP ERROR
11116                                                                  ;NO GO TO ERROR
11117  054402  077105                  5$:     SOB     R1,4$                   ;ARE WE DONE
11118  054404  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11119  054410  001401                          BEQ     6$                      ;OK GO ON
11120  054412  104003                          ERROR   +3              ;FPP ERROR
11121                                                                  ;NO GO TO ERROR
11122  054414  012637  000004          6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11123
11124                                  ;
11125  054420  000167  000004                  JMP     FIN16
11126                                  ;
11127  054424  104003                  TSF16:  ERROR   +3              ;FPP ERROR
11128                                                                  ;ODD ADDRESS TRAP
11129  054426  000006                          RTT                             ;RETURN
11130                                  ;
11131  054430  000240                  FIN16:  NOP
11132                                  ;
11133                                  ;
11134                                  ;---------------------------------
11135                                  ;TEST  FDST SOP MODE 5
11136                                  ;
11137                                  ;
11138  054432                          TSFP17:
11139                                  ;
11140  054432  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11141  054436  012737  054606  000004          MOV     #TSF17,@#4              ;SETUP NEW VECTOR
11142  054444  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11143  054450  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11144  054454  170102                          LDFPS   R2                      ;SET FD=1
11145  054456  012705  000011                  MOV     #9.,R5                  ;INIT COUNTER
11146  054462  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11147  054466  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11148  054472  077503                          SOB     R5,100$                 ;ARE WE DONE
11149  054474  012737  003134  003122          MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11150  054502  012702  003124                  MOV     #TSTLOC+2,R2                    ;SETUP POINTER TO DATA
11151  054506  170452                          CLRD    @-(R2)                  ; TEST INSTRUCTION
11152  054510  170203                          STFPS   R3                      ;GET FPS
11153  054512  020227  003122                  CMP     R2,#TSTLOC                      ;IS R2 CORRECT
11154  054516  001401                          BEQ     1$                      ;YES GO ON
11155  054520  104003                          ERROR   +3              ;FPP ERROR
11156                                                                  ;NO GO TO ERROR
11157  054522  022227  003134          1$:     CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11158  054526  001401                          BEQ     2$                      ;YES GO ON
11159  054530  104003                          ERROR   +3              ;FPP ERROR
```

```
11160
11161  054532  012701  000004          2$:     MOV     #4,R1                   ;NO GO TO ERROR
11162  054536  022227  177777          3$:     CMP     (R2)+,#177777                   ;INIT COUNTER
11163  054542  001401                          BEQ     4$                              ;IS LOCATION ALL ONES
11164  054544  104003                          ERROR   .3                              ;YES GO ON
11165                                                                           ;FPP ERROR
11166  054546  077105                  4$:     SOB     R1,3$                   ;NO GO TO ERROR
11167  054550  012701  000004                  MOV     #4,R1                           ;ARE WE DONE
11168  054554  022227  000000          5$:     CMP     (R2)+,#0                        ;INIT COUNTER
11169  054560  001401                          BEQ     6$                              ;IS LOCATION 0
11170  054562  104003                          ERROR   .3                              ;YES GO ON
11171                                                                           ;FPP ERROR
11172  054564  077105                  6$:     SOB     R1,5$                   ;NO GO TO ERROR
11173  054566  020327  000204                  CMP     R3,#204                         ;ARE WE DONE
11174  054572  001401                          BEQ     7$                              ;CHECK FPS
11175  054574  104003                          ERROR   .3                              ;OK GO ON
11176                                                                           ;FPP ERROR
11177  054576  012637  000004          7$:     MOV     (SP)+,@#4               ;NO GO TO ERROR
11178                                                                                   ;RESTORE VECTOR
11179                                  ;
11180  054602  000167  000004                  JMP     FIN17
11181                                  ;
11182  054606  104003                  TSF17:  ERROR   .3                      ;FPP ERROR
11183                                                                           ;ODD ADDRESS TRAP
11184  054610  000006                          RTT                                     ;RETURN
11185                                  ;
11186  054612  000240                  FIN17:  NOP
11187                                  ;
11188                                  ;
11189                                  ;---------------------------------
11190                                  ;TEST  FDST SOP MODE 6
11191                                  ;
11192                                  ;
11193  054614                          TSFP20:
11194                                  ;
11195  054614  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11196  054620  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11197  054624  012737  054760  000004          MOV     #TSF20,@#4              ;SETUP NEW VECTOR
11198  054632  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11199  054636  170102                          LDFPS   R2                      ;SET FD=1
11200  054640  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
11201  054644  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11202  054650  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11203  054654  077503                          SOB     R5,100$                 ;ARE WE DONE
11204  054656  012702  003123                  MOV     #TSTLOC+1,R2                    ;SETUP POINTER TO DATA
11205  054662  170462  000007                  CLRD    7(R2)                   ; TEST INSTRUCTION
11206  054666  170203                          STFPS   R3                      ;GET FPS
11207  054670  020227  003123                  CMP     R2,#TSTLOC+1                    ;IS R2 CORRECT
11208  054674  001401                          BEQ     1$                      ;YES GO ON
11209  054676  104003                          ERROR   .3                      ;FPP ERROR
11210                                                                           ;NO GO TO ERROR
11211  054700  012702  003122          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11212  054704  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11213  054710  022227  177777          2$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11214  054714  001401                          BEQ     3$                      ;YES GO ON
11215  054716  104003                          ERROR   .3                      ;FPP ERROR
```

```
11216
11217  054720  077105                  3$:     SOB     R1,2$                   ;NO GO TO ERROR
11218  054722  012701  000004                  MOV     #4,R1                           ;ARE WE DONE
11219  054726  022227  000000          4$:     CMP     (R2)+,#0                        ;INIT COUNTER
11220  054732  001401                          BEQ     5$                              ;IS DATA CORRECT
11221  054734  104003                          ERROR   +3                              ;YES GO ON
11222                                                                          ;FPP ERROR
11223  054736  077105                  5$:     SOB     R1,4$                   ;NO GO TO ERROR
11224  054740  020327  000204                  CMP     R3,#204                         ;ARE WE DONE
11225  054744  001401                          BEQ     6$                              ;IS FPS CORRECT
11226  054746  104003                          ERROR   +3                              ;YES GO ON
11227                                                                          ;FPP ERROR
11228  054750  012637  000004          6$:     MOV     (SP)+,@#4               ;NO GO TO ERROR
11229                                                                                  ;RESTORE VECTOR
11230                                  ;
11231  054754  000167  000004                  JMP     FIN20
11232                                  ;
11233  054760  104003                  TSF20:  ERROR   +3                      ;FPP ERROR
11234                                                                          ;ODD ADDRESS TRAP
11235  054762  000006                          RTT                                     ;RETURN
11236                                  ;
11237  054764  000240                  FIN20:  NOP
11238                                  ;
11239                                  ;
11240                                  ;---------------------------------
11241                                  ;TEST  FDST SOP MODE 7
11242                                  ;
11243                                  ;
11244  054766                          TSFP21:
11245                                  ;
11246  054766  005037  132646                  CLR     @#TRPFLG                        ;CLEAR TRAP FLAG
11247  054772  013746  000004                  MOV     @#4,-(SP)                       ;SAVE TIMEOUT VECTOR
11248  054776  012737  055150  000004          MOV     #TSF21,@#4                      ;SETUP NEW VECTOR
11249  055004  012702  000200                  MOV     #200,R2                         ;SETUP TO LOAD FPS
11250  055010  170102                          LDFPS   R2                              ;SET FD=1
11251  055012  012705  000010                  MOV     #8.,R5                          ;INIT COUNTER
11252  055016  012704  003122                  MOV     #TSTLOC,R4                              ;SETUP POINTER TO TEST LOCATION
11253  055022  012724  177777          100$:   MOV     #177777,(R4)+   ;INIT TEST LOCATION
11254  055026  077503                          SOB     R5,100$                 ;ARE WE DONE
11255  055030  012737  003122  003132          MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11256  055036  012702  003125                  MOV     #TSTLOC+3,R2                    ;SETUP POINTER TO DATA
11257  055042  170472  000005                  CLRD    @5(R2)                  ; TEST INSTRUCTION
11258  055046  170203                          STFPS   R3                              ;GET FPS
11259  055050  020227  003125                  CMP     R2,#TSTLOC+3                            ;IS R2 CORRECT
11260  055054  001401                          BEQ     1$                              ;YES GO ON
11261  055056  104003                          ERROR   +3                      ;FPP ERROR
11262                                                                          ;NO GO TO ERROR
11263  055060  012702  003122          1$:     MOV     #TSTLOC,R2                              ;SETUP POINTER TO DATA
11264  055064  012701  000004                  MOV     #4,R1                           ;INIT COUNTER
11265  055070  022227  000000          2$:     CMP     (R2)+,#0                        ;IS DATA CORRECT
11266  055074  001401                          BEQ     3$                              ;YES GO ON
11267  055076  104003                          ERROR   +3                      ;FPP ERROR
11268                                                                          ;NO GO TO ERROR
11269  055100  077105                  3$:     SOB     R1,2$                           ;ARE WE DONE
11270  055102  022227  003122                  CMP     (R2)+,#TSTLOC                   ;IS DATA CORRECT
11271  055106  001401                          BEQ     4$                              ;YES GO ON
```

```
11272  055110  104003                          ERROR   .3              ;FPP ERROR
11273                                                                   ;NO GO TO ERROR
11274  055112  012701  000003          4$:     MOV     #3,R1                   ;INIT COUNTER
11275  055116  022227  177777          5$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11276  055122  001401                          BEQ     6$                      ;YES GO ON
11277  055124  104003                          ERROR   .3              ;FPP ERROR
11278                                                                   ;NO GO TO ERROR
11279  055126  077105                  6$:     SOB     R1,5$                   ;ARE WE DONE
11280  055130  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11281  055134  001401                          BEQ     7$                      ;OK GO ON
11282  055136  104003                          ERROR   .3              ;FPP ERROR
11283                                                                   ;NO GO TO ERROR
11284  055140  012637  000004          7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11285
11286                                  ;
11287  055144  000167  000004                  JMP     FIN21
11288                                  ;
11289  055150  104003                  TSF21:  ERROR   .3              ;FPP ERROR
11290                                                                   ;ODD ADDRESS TRAP
11291  055152  000006                          RTT                             ;RETURN
11292                                  ;
11293  055154  000240                  FIN21:  NOP
11294                                  ;
11295                                  ;
11296                                  ;---------------------------------
11297                                  ;TEST  FDST SOP MODE 3 GR7
11298                                  ;
11299                                  ;
11300  055156                          TSFP22:
11301                                  ;
11302  055156  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11303  055162  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
11304  055166  012737  055314  000004          MOV     #TSF22,@#4              ;SETUP NEW VECTOR
11305  055174  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11306  055200  170102                          LDFPS   R2                      ;SET FD=1
11307  055202  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
11308  055206  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11309  055212  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11310  055216  077503                          SOB     R5,100$                 ;ARE WE DONE
11311  055220  012737  003132  003122          MOV     #TSTLOC+10,@#TSTLOC             ;INIT TEST LOCATION
11312  055226  170437  003122                  CLRD    @#TSTLOC                        ; TEST INSTRUCTION
11313  055232  170203                          STFPS   R3                      ;GET FPS
11314  055234  012702  003122                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11315  055240  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11316  055244  022227  000000          1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11317  055250  001401                          BEQ     2$                      ;YES GO ON
11318  055252  104003                          ERROR   .3              ;FPP ERROR
11319                                                                   ;NO GO TO ERROR
11320  055254  077105                  2$:     SOB     R1,1$                   ;ARE WE DONE
11321  055256  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11322  055262  022227  177777          3$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11323  055266  001401                          BEQ     4$                      ;YES GO ON
11324  055270  104003                          ERROR   .3              ;FPP ERROR
11325                                                                   ;NO GO TO ERROR
11326  055272  077105                  4$:     SOB     R1,3$                   ;ARE WE DONE
11327  055274  020327  000204                  CMP     R3,#204                 ;CHECK FPS
```

```
11328  055300  001401                    BEQ     5$                      ;OK GO ON
11329  055302  104003                    ERROR   .3              ;FPP ERROR
11330                                                             ;NO GO TO ERROR
11331  055304  012637  000004    5$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11332
11333                            ;
11334  055310  000167  000004            JMP     FIN22
11335                            ;
11336  055314  104003            TSF22:  ERROR   .3              ;FPP ERROR
11337                                                             ;ODD ADDRESS TRAP
11338  055316  000006                    RTT                     ;RETURN
11339                            ;
11340  055320  000240            FIN22:  NOP
11341                            ;
11342                            ;
11343                            ;--------------------------------
11344                            ;TEST  FDST SOP MODE 6 GR7
11345                            ;
11346                            ;
11347  055322                    TSFP23:
11348                            ;
11349  055322  005037  132646            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11350  055326  013746  000004            MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11351  055332  012737  055434  000004    MOV     #TSF23,@#4              ;SETUP NEW VECTOR
11352  055340  012702  000200            MOV     #200,R2                 ;SETUP TO LOAD FPS
11353  055344  170102                    LDFPS   R2                      ;SET FD=1
11354  055346  012705  000004            MOV     #4,R5                   ;INIT COUNTER
11355  055352  012704  003122            MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11356  055356  012724  177777    100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11357  055362  077503                    SOB     R5,100$                 ;ARE WE DONE
11358  055364  170467  125332            CLRD    TSTLOC                  ; TEST INSTRUCTION
11359  055370  170203                    STFPS   R3                      ;GET FPS
11360  055372  012701  000004            MOV     #4,R1                   ;INIT COUNTER
11361  055376  012702  003122            MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11362  055402  022227  000000    1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11363  055406  001401                    BEQ     2$                      ;YES GO ON
11364  055410  104003                    ERROR   .3              ;FPP ERROR
11365                                                             ;NO GO TO ERROR
11366  055412  077105            2$:     SOB     R1,1$                   ;ARE WE DONE
11367  055414  020327  000204            CMP     R3,#204                 ;CHECK FPS
11368  055420  001401                    BEQ     3$                      ;OK GO ON
11369  055422  104003                    ERROR   .3              ;FPP ERROR
11370                                                             ;NO GO TO ERROR
11371  055424  012637  000004    3$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11372
11373                            ;
11374  055430  000167  000004            JMP     FIN23
11375                            ;
11376  055434  104003            TSF23:  ERROR   .3              ;FPP ERROR
11377                                                             ;ODD ADDRESS TRAP
11378  055436  000006                    RTT                     ;
11379  055440  000240            FIN23:  NOP
11380                            ;
11381                            ;
11382                            ;--------------------------------
11383                            ;TEST  FDST SOP MODE 7 GR7
```

```
11384                                           ;
11385                                           ;
11386  055442                                   TSFP24:
11387                                           ;
11388  055442  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11389  055446  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11390  055452  012737  055610  000004          MOV     #TSF24,@#4              ;SETUP NEW VECTOR
11391  055460  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11392  055464  170102                          LDFPS   R2                      ;SET FD=1
11393  055466  012705  000010                  MOV     #8.,R5          ;INIT COUNTER
11394  055472  012704  003122                  MOV     #TSTLOC,R4              ;SETUP TEST LOCATION POINTER
11395  055476  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11396  055502  077503                          SOB     R5,100$                 ;ARE WE DONE
11397  055504  012737  003122  003132          MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11398  055512  170477  125414                  CLRD    @TSTLOC+10                      ; TEST INSTRUCTION
11399  055516  170203                          STFPS   R3                      ;GET FPS
11400  055520  012702  003122                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11401  055524  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11402  055530  022227  000000          1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11403  055534  001401                          BEQ     2$                      ;YES GO ON
11404  055536  104003                          ERROR   .+3             ;FPP ERROR
11405                                           ;NO GO TO ERROR
11406  055540  077105          2$:     SOB     R1,1$                   ;ARE WE DONE
11407  055542  022227  003122                  CMP     (R2)+,#TSTLOC           ;IS DATA CORRECT
11408  055546  001401                          BEQ     3$                      ;YES GO ON
11409  055550  104003                          ERROR   .+3             ;FPP ERROR
11410                                           ;NO GO TO ERROR
11411  055552  012701  000003          3$:     MOV     #3,R1                   ;INIT COUNTER
11412  055556  022227  177777          4$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11413  055562  001401                          BEQ     5$                      ;YES GO ON
11414  055564  104003                          ERPOR   .+3             ;FPP ERROR
11415                                           ;NO GO TO ERROR
11416  055566  077105          5$:     SOB     R1,4$                   ;ARE WE DONE
11417  055570  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11418  055574  001401                          BEQ     6$                      ;OK GO ON
11419  055576  104003                          ERROR   .+3             ;FPP ERROR
11420                                           ;NO GO TO ERROR
11421  055600  012637  000004          6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11422
11423                                           ;
11424  055604  000167  000004                  JMP     FIN24
11425                                           ;
11426  055610  104003                  TSF24:  ERROR   .+3             ;FPP ERROR
11427                                           ;ODD ADDRESS TRAP
11428  055612  000006                          RTT                             ;
11429                                           ;
11430  055614  000240                  FIN24:  NOP
11431                                           ;
11432                                           ;
11433                                           ;---------------------------------
11434                                           ;TEST  CLRF
11435                                           ;
11436                                           ;
11437  055616                                   TSFP25:
11438                                           ;
11439  055616  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
```

```
11440  055622  005002                          CLR     R2                      ;SETUP TO LOAD FPS
11441  055624  170102                          LDFPS   R2                      ;SET FD=0
11442  055626  012705  000004                  MOV     #4,R5                   ;INIT COUNTER
11443  055632  012704  003122                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11444  055636  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11445  055642  077503                          SOB     R5,100$                 ;ARE WE DONE
11446  055644  012702  003122                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11447  055650  170422                          CLRF    (R2)+                   ; TEST INSTRUCTION
11448  055652  170203                          STFPS   R3                      ;GET FPS
11449  055654  020227  003126                  CMP     R2,#TSTLOC+4                    ;IS R2 CORRECT
11450  055660  001401                          BEQ     1$                      ;YES GO ON
11451  055662  104003                          ERROR   ;3                      ;FPP ERROR
11452                                                                          ;NO GO TO ERROR
11453  055664  012702  003122          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11454  055670  012701  000002                  MOV     #2,R1                   ;INIT COUNTER
11455  055674  022227  000000          2$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11456  055700  001401                          BEQ     3$                      ;YES GO ON
11457  055702  104003                          ERROR   ;3                      ;FPP ERROR
11458                                                                          ;NO GO TO ERROR
11459  055704  077105                  3$:     SOB     R1,2$                   ;ARE WE DONE
11460  055706  012701  000002                  MOV     #2,R1                   ;INIT COUNTER
11461  055712  022227  177777          4$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11462  055716  001401                          BEQ     5$                      ;YES GO ON
11463  055720  104003                          ERROR   ;3                      ;FPP ERROR
11464                                                                          ;NO GO TO ERROR
11465  055722  077105                  5$:     SOB     R1,4$                   ;ARE WE DONE
11466  055724  020327  000004                  CMP     R3,#4                   ;CHECK FPS
11467  055730  001401                          BEQ     6$                      ;OK GO ON
11468  055732  104003                          ERROR   ;3                      ;FPP ERROR
11469                                                                          ;NO GO TO ERROR
11470  055734                          6$:
11471                                  ;
11472                                  ;
11473                                  ;-------------------- -------
11474                                  ;TEST  TSTF AND TSTD
11475                                  ;
11476                                  ;
11477  055734                          TSFP26:
11478                                  ;
11479  055734  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11480  055740  005004                          CLR     R4                      ;SETUP TO LOAD FPS
11481  055742  170104                          LDFPS   R4                      ;SET FD=0
11482  055744  170567  000244                  TSTF    TS26D0                  ; TEST INSTRUCTION
11483  055750  170203                          STFPS   R3                      ;GET FPS
11484  055752  020327  000004                  CMP     R3,#4                   ;CHECK FPS
11485  055756  001401                          BEQ     1$                      ;OK GO ON
11486  055760  104003                          ERROR   ;3                      ;FPP ERROR
11487                                                                          ;NO GO TO ERROR
11488  055762  012704  056214          1$:     MOV     #TS26D0,R4                      ;SETUP POINTERS TO DATA
11489  055766  012702  056244                  MOV     #TS26D3,R2              ;
11490  055772  004767  000200                  JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11491  055776  170537  056224                  TSTF    @#TS26D1                ; TEST INSTRUCTION
11492  056002  170203                          STFPS   R3                      ;GET FPS
11493  056004  020327  000010                  CMP     R3,#10                  ;CHECK FPS
11494  056010  001401                          BEQ     2$                      ;OK GO ON
11495  056012  104003                          ERROR   ;3                      ;FPP ERROR
```

```
11496
11497  056014  012704  056224          2$:     MOV     #TS26D1,R4      ;NO GO TO ERROR
11498  056020  012702  056254                  MOV     #TS26D4,R2              ;SETUP POINTERS TO DATA
11499  056024  004767  000146                  JSR     PC,CHEC26               ;
11500  056030  170567  000200                  TSTF    TS26D2                  ;CHECK IF DATA IS CORRECT
11501  056034  170203                          STFPS   R3                      ; TEST INSTRUCTION
11502  056036  020327  000000                  CMP     R3,#0                   ;GET FPS
11503  056042  001401                          BEQ     3$                      ;CHECK FPS
11504  056044  104003                          ERROR   .3                      ;OK GO ON
11505                                                                          ;FPP ERROR
11506  056046  012704  056234          3$:     MOV     #TS26D2,R4      ;NO GO TO ERROR
11507  056052  012702  056264                  MOV     #TS26D5,R2              ;SETUP POINTERS TO DATA
11508  056056  004767  000114                  JSR     PC,CHEC26               ;
11509  056062  012704  000200                  MOV     #200,R4                 ;CHECK IF DATA IS CORRECT
11510  056066  170104                          LDFPS   R4                      ;SETUP TO LOAD FPS
11511  056070  170537  056214                  TSTD    @#TS26D0                ;SET FD=1
11512  056074  170203                          STFPS   R3                      ; TEST INSTRUCTION
11513  056076  020327  000204                  CMP     R3,#204                 ;GET FPS
11514  056102  001401                          BEQ     4$                      ;CHECK FPS
11515  056104  104003                          ERROR   .3                      ;OK GO ON
11516                                                                          ;FPP ERROR
11517  056106  012704  056214          4$:     MOV     #TS26D0,R4      ;NO GO TO ERROR
11518  056112  012702  056244                  MOV     #TS26D3,R2              ;SETUP POINTERS TO DATA
11519  056116  004767  000054                  JSR     PC,CHEC26               ;
11520  056122  170567  000076                  TSTD    TS26D1                  ;CHECK IF DATA IS CORRECT
11521  056126  170203                          STFPS   R3                      ; TEST INSTRUCTION
11522  056130  020327  000210                  CMP     R3,#210                 ;GET FPS
11523  056134  001401                          BEQ     5$                      ;CHECK FPS
11524  056136  104003                          ERROR   .3                      ;OK GO ON
11525                                                                          ;FPP ERROR
11526  056140  012704  056224          5$:     MOV     #TS26D1,R4      ;NO GO TO ERROR
11527  056144  012702  056254                  MOV     #TS26D4,R2              ;SETUP POINTERS TO DATA
11528  056150  004767  000022                  JSR     PC,CHEC26               ;
11529  056154  170567  000054                  TSTD    TS26D2                  ;CHECK IF DATA IS CORRECT
11530  056160  170203                          STFPS   R3                      ; TEST INSTRUCTION
11531  056162  020327  000200                  CMP     R3,#200                 ;GET FPS
11532  056166  001401                          BEQ     6$                      ;CHECK FPS
11533  056170  104003                          ERROR   .3                      ;OK GO ON
11534                                                                          ;FPP ERROR
11535  056172                          6$:                             ;NO GO TO ERROR
11536                                  ;
11537  056172  000167  000076                  JMP     FIN26
11538                                  ;
11539  056176  012701  000004          CHEC26: MOV     #4,R1                   ;INIT COUNTER
11540  056202  022422                  1$:     CMP     (R4)+,(R2)+             ;IS DATA CORRECT
11541  056204  001401                          BEQ     2$                      ;YES GO ON
11542  056206  104003                          ERROR   .3                      ;FPP ERROR
11543                                                                          ;NO GO TO ERROR
11544  056210  077104                  2$:     SOB     R1,1$                   ;ARE WE DONE
11545  056212  000207                          RTS     PC                      ;RETURN
11546
11547  056214  000177          TS26D0: .WORD   177
11548  056216  177777                  .WORD   177777
11549  056220  177777                  .WORD   177777
11550  056222  177777                  .WORD   177777
11551  056224  177777          TS26D1: .WORD   177777
```

```
11552  056226  000000                          .WORD   0
11553  056230  000000                          .WORD   0
11554  056232  000000                          .WORD   0
11555  056234  077777          TS26D2: .WORD   77777
11556  056236  000000                          .WORD   0
11557  056240  000000                          .WORD   0
11558  056242  000000                          .WORD   0
11559  056244  000177          TS26D3: .WORD   177
11560  056246  177777                          .WORD   177777
11561  056250  177777                          .WORD   177777
11562  056252  177777                          .WORD   177777
11563  056254  177777          TS26D4: .WORD   177777
11564  056256  000000                          .WORD   0
11565  056260  000000                          .WORD   0
11566  056262  000000                          .WORD   0
11567  056264  077777          TS26D5: .WORD   77777
11568  056266  000000                          .WORD   0
11569  056270  000000                          .WORD   0
11570  056272  000000                          .WORD   0
11571  056274  000240          FIN26:  NOP
11572                          ;
11573                          ;
11574                          ;---------------------------------
11575                          ;TEST  ABSF
11576                          ;
11577                          ;
11578  056276                  TSFP27:
11579                          ;
11580  056276  005037  132646          CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11581  056302  005005                   CLR     R5                      ;SETUP TO LOAD FPS
11582  056304  170105                   LDFPS   R5                      ;SET FD=0
11583  056306  012701  000014          MOV     #12.,R1                 ;INIT COUNTER
11584  056312  012704  003122          MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11585  056316  012703  056522          MOV     #TS27D0,R3              ;SETUP POINTER TO TEST VALUE
11586  056322  012324          100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11587  056324  077102                   SOB     R1,100$                 ;ARE WE DONE
11588  056326  012705  003122          MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11589  056332  170615                   ABSF    (R5)                    ; TEST INSTRUCTION
11590  056334  170203                   STFPS   R3                      ;GET FPS
11591  056336  020527  003122          CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11592  056342  001401                   BEQ     1$                      ;YES GO ON
11593  056344  104003                   ERROR   +3                      ;FPP ERROR
11594                                                                   ;NO GO TO ERROR
11595  056346  012702  056552  1$:      MOV     #TS27D3,R2              ;SETUP POINTER TO DATA
11596  056352  004767  000126          JSR     PC,CHEC27               ;CHECK IF DATA IS CORRECT
11597  056356  020327  000000          CMP     R3,#0                   ;CHECK FPS
11598  056362  001401                   BEQ     2$                      ;OK GO ON
11599  056364  104003                   ERROR   +3                      ;FPP ERROR
11600                                                                   ;NO GO TO ERROR
11601  056366  012705  003132  2$:      MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
11602  056372  170625                   ABSF    (R5)+                   ; TEST INSTRUCTION
11603  056374  170203                   STFPS   R3                      ;GET FPS
11604  056376  020527  003136          CMP     R5,#TSTLOC+14                   ;IS R5 CORRECT
11605  056402  001401                   BEQ     3$                      ;YES GO ON
11606  056404  104003                   ERROR   +3                      ;FPP ERROR
11607                                                                   ;NO GO TO ERROR
```

```
11608  056406  012705  003132          3$:     MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
11609  056412  012702  056562                  MOV     #TS27D4,R2                      ;
11610  056416  004767  000062                  JSR     PC,CHEC27                       ;CHECK IF DATA IS CORRECT
11611  056422  020327  000000                  CMP     R3,#0                           ;CHECK FPS
11612  056426  001401                          BEQ     4$                              ;OK GO ON
11613  056430  104003                          ERROR   .3                      ;FPP ERROR
11614                                                                          ;NO GO TO ERROR
11615  056432  012705  003122          4$:     MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11616  056436  170665  000020                  ABSF    20(R5)                  ; TEST INSTRUCTION
11617  056442  170203                          STFPS   R3                      ;GET FPS
11618  056444  020527  003122                  CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11619  056450  001401                          BEQ     5$                              ;YES GO ON
11620  056452  104003                          ERROR   .3                      ;FPP ERROR
11621                                                                          ;NO GO TO ERROR
11622  056454  012705  003142          5$:     MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
11623  056460  012702  056572                  MOV     #TS27D5,R2                      ;
11624  056464  004767  000014                  JSR     PC,CHEC27                       ;CHECK IF DATA IS CORRECT
11625  056470  020327  000004                  CMP     R3,#4                           ;CHECK FPS
11626  056474  001401                          BEQ     6$                              ;OK GO ON
11627  056476  104003                          ERROR   .3                      ;FPP ERROR
11628                                                                          ;NO GO TO ERROR
11629  056500                          6$:
11630                                  ;
11631  056500  000167  000076                  JMP     FIN27
11632                                  ;
11633  056504  012701  000004          CHEC27: MOV     #4,R1                           ;INIT COUNTER
11634  056510  022522                  1$:     CMP     (R5)+,(R2)+                     ;IS DATA CORRECT
11635  056512  001401                          BEQ     2$                              ;YES GO ON
11636  056514  104003                          ERROR   .3                      ;FPP ERROR
11637                                                                          ;NO GO TO ERROR
11638  056516  077104                  2$:     SOB     R1,1$                   ;ARE WE DONE
11639  056520  000207                          RTS     PC                              ;RETURN
11640                                  ;
11641  056522  177777          TS27D0: .WORD   177777
11642  056524  177777                  .WORD   177777
11643  056526  177777                  .WORD   177777
11644  056530  177777                  .WORD   177777
11645  056532  000377          TS27D1: .WORD   377
11646  056534  175436                  .WORD   175436
11647  056536  136477                  .WORD   136477
11648  056540  000001                  .WORD   1
11649  056542  000177          TS27D2: .WORD   177
11650  056544  175436                  .WORD   175436
11651  056546  136477                  .WORD   136477
11652  056550  000001                  .WORD   1
11653  056552  077777          TS27D3: .WORD   77777
11654  056554  177777                  .WORD   177777
11655  056556  177777                  .WORD   177777
11656  056560  177777                  .WORD   177777
11657  056562  000377          TS27D4: .WORD   377
11658  056564  175436                  .WORD   175436
11659  056566  136477                  .WORD   136477
11660  056570  000001                  .WORD   1
11661  056572  000000          TS27D5: .WORD   0
11662  056574  000000                  .WORD   0
11663  056576  136477                  .WORD   136477
```

```
11664  056600  000001                          .WORD   1
11665  056602  000240                  FIN27:  NOP
11666                                  ;
11667                                  ;
11668                                  ;-------------------------------
11669                                  ;TEST  ABSD
11670                                  ;
11671                                  ;
11672  056604                          TSFP30:
11673                                  ;
11674  056604  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11675  056610  012705  000200                  MOV     #200,R5                 ;SETUP TO LOAD FPS
11676  056614  170105                          LDFPS   R5                      ;SET FD=1
11677  056616  012701  000014                  MOV     #12.,R1                 ;INIT COUNTER
11678  056622  012704  003122                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11679  056626  012703  057032                  MOV     #TS3000,R3              ;SETUP POINTER TO TEST VALUE
11680  056632  012324                  100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11681  056634  077102                          SOB     R1,100$                 ;ARE WE DONE
11682  056636  012705  003122                  MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11683  056642  170615                          ABSD    (R5)                    ; TEST INSTRUCTION
11684  056644  170203                          STFPS   R3                      ;GET FPS
11685  056646  020527  003122                  CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11686  056652  001401                          BEQ     1$                      ;YES GO ON
11687  056654  104003                          ERROR   .+3                     ;FPP ERROR
11688                                                                          ;NO GO TO ERROR
11689  056656  012702  057062          1$:     MOV     #TS3003,R2              ;SETUP POINTER TO DATA
11690  056662  004767  000126                  JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11691  056666  020327  000200                  CMP     R3,#200                 ;CHECK FPS
11692  056672  001401                          BEQ     2$                      ;OK GO ON
11693  056674  104003                          ERROR   .+3                     ;FPP ERROR
11694                                                                          ;NO GO TO ERROR
11695  056676  012705  003132          2$:     MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
11696  056702  170625                          ABSD    (R5)+                   ; TEST INSTRUCTION
11697  056704  170203                          STFPS   R3                      ;GET FPS
11698  056706  020527  003142                  CMP     R5,#TSTLOC+20                   ;IS R5 CORRECT
11699  056712  001401                          BEQ     3$                      ;YES GO ON
11700  056714  104003                          ERROR   .+3                     ;FPP ERROR
11701                                                                          ;NO GO TO ERROR
11702  056716  012705  003132          3$:     MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
11703  056722  012702  057072                  MOV     #TS3004,R2              ;
11704  056726  004767  000062                  JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11705  056732  020327  000200                  CMP     R3,#200                 ;CHECK FPS
11706  056736  001401                          BEQ     4$                      ;OK GO ON
11707  056740  104003                          ERROR   .+3                     ;FPP ERROR
11708                                                                          ;NO GO TO ERROR
11709  056742  012705  003122          4$:     MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11710  056746  170665  000020                  ABSD    20(R5)                  ; TEST INSTRUCTION
11711  056752  170203                          STFPS   R3                      ;GET FPS
11712  056754  020527  003122                  CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11713  056760  001401                          BEQ     5$                      ;YES GO ON
11714  056762  104003                          ERROR   .+3                     ;FPP ERROR
11715                                                                          ;NO GO TO ERROR
11716  056764  012705  003142          5$:     MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
11717  056770  012702  057102                  MOV     #TS3005,R2              ;
11718  056774  004767  000014                  JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11719  057000  020327  000204                  CMP     R3,#204                 ;CHECK FPS
```

```
11720  057004  001401                  BEQ     6$                      ;OK GO ON
11721  057006  104003                  ERROR   .3                      ;FPP ERROR
11722                                                                   ;NO GO TO ERROR
11723  057010                          6$:
11724                                  ;
11725  057010  000167  000076          JMP     FIN30
11726                                  ;
11727  057014  012701  000004          CHEC30: MOV     #4,R1                   ;INIT COUNTER
11728  057020  022522                  1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
11729  057022  001401                          BEQ     2$                      ;YES GO ON
11730  057024  104003                          ERROR   .3                      ;FPP ERROR
11731                                                                           ;NO GO TO ERROR
11732  057026  077104                  2$:     SOB     R1,1$                   ;ARE WE DONE
11733  057030  000207                          RTS     PC                      ;RETURN
11734                                  ;
11735  057032  177777                  TS30D0: .WORD   177777
11736  057034  177777                          .WORD   177777
11737  057036  177777                          .WORD   177777
11738  057040  177777                          .WORD   177777
11739  057042  000377                  TS30D1: .WORD   377
11740  057044  175436                          .WORD   175436
11741  057046  136477                          .WORD   136477
11742  057050  000001                          .WORD   1
11743  057052  000177                  TS30D2: .WORD   177
11744  057054  175436                          .WORD   175436
11745  057056  136477                          .WORD   136477
11746  057060  000001                          .WORD   1
11747  057062  077777                  TS30D3: .WORD   77777
11748  057064  177777                          .WORD   177777
11749  057066  177777                          .WORD   177777
11750  057070  177777                          .WORD   177777
11751  057072  000377                  TS30D4: .WORD   377
11752  057074  175436                          .WORD   175436
11753  057076  136477                          .WORD   136477
11754  057100  000001                          .WORD   1
11755  057102  000000                  TS30D5: .WORD   0
11756  057104  000000                          .WORD   0
11757  057106  000000                          .WORD   0
11758  057110  000000                          .WORD   0
11759  057112  000240                  FIN30:  NOP
11760                                  ;
11761                                  ;
11762                                  ;---------------------------------
11763                                  ;TEST  FDST SOP MODE 2 GR7
11764                                  ;
11765                                  ;
11766  057114                          TSFP31:
11767                                  ;
11768  057114  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11769  057120  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11770  057124  012737  057226  000004          MOV     #TSF31,@#4              ;SETUP NEW VECTOR
11771  057132  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11772  057136  170102                          LDFPS   R2                      ;SET FD=1
11773  057140  170527  000005          TSD31:  TSTD    #5                      ; TEST INSTRUCTION
11774  057144  000240                          NOP
11775  057146  000240                          NOP
```

```
11776  057150  000240                          NOP
11777  057152  170203                          STFPS   R3                      ;GET FPS
11778  057154  020327  000204                  CMP     R3,#0204                ;CHECK FPS
11779  057160  001401                          BEQ     1$                      ;OK GO ON
11780  057162  104003                          ERROR   .3              ;FPP ERROR
11781                                                                  ;NO GO TO ERROR
11782  057164  012702  057142          1$:     MOV     #TSD31+2,R2             ;SETUP POINTER TO DATA
11783  057170  022227  000005                  CMP     (R2)+,#5                ;IS DATA CORRECT
11784  057174  001401                          BEQ     2$                      ;YES GO ON
11785  057176  104003                          ERROR   .3              ;FPP ERROR
11786                                                                  ;NO GO TO ERROR
11787  057200  012701  000003          2$:     MOV     #3,R1                   ;INIT COUNTER
11788  057204  022227  000240          3$:     CMP     (R2)+,#240              ;IS DATA CORRECT
11789  057210  001401                          BEQ     4$                      ;YES GO ON
11790  057212  104003                          ERROR   .3              ;FPP ERROR
11791                                                                  ;NO GO TO ERROR
11792  057214  077105          4$:     SOB     R1,3$                   ;ARE WE DONE
11793  057216  012637  000004                  MOV     (SP)+,@#4               ;RESTORE VECTOR
11794
11795                                  ;
11796  057222  000167  000004                  JMP     FIN31
11797                                  ;
11798  057226  104003                  TSF31:  ERROR   .3              ;FPP ERROR
11799                                                                  ;ODD ADDRESS TRAP
11800  057230  000006                          RTT                             ;RETURN
11801                                  ;
11802  057232  000240                  FIN31:  NOP
11803                                  ;
11804                                  ;
11805                                  ;-------------------------------
11806                                  ;TEST  NEGF
11807                                  ;
11808                                  ;
11809  057234                          TSFP32:
11810                                  ;
11811  057234  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11812  057240  005005                          CLR     R5                      ;SETUP TO LOAD FPS
11813  057242  170105                          LDFPS   R5                      ;SET FD=0
11814  057244  012701  000014                  MOV     #12.,R1                 ;INIT COUNTER
11815  057250  012704  003122                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11816  057254  012703  057424                  MOV     #TS32D0,R3              ;SETUP POINTER TO TEST VALUE
11817  057260  012324                  100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11818  057262  077102                          SOB     R1,100$                 ;ARE WE DONE
11819  057264  170767  123632                  NEGF    TSTLOC                  ; TEST INSTRUCTION
11820  057270  170203                          STFPS   R3                      ;GET FPS
11821  057272  012705  003122                  MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
11822  057276  012702  057454                  MOV     #TS32D3,R2              ;
11823  057302  004767  000100                  JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
11824  057306  020327  000000                  CMP     R3,#0                   ;CHECK FPS
11825  057312  001401                          BEQ     1$                      ;YES GO ON
11826  057314  104003                          ERROR   .3              ;FPP ERROR
11827                                                                  ;NO GO TO ERROR
11828  057316  170767  123610          1$:     NEGF    TSTLOC+10                       ; TEST INSTRUCTION
11829  057322  170203                          STFPS   R3                      ;GET FPS
11830  057324  012705  003132                  MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
11831  057330  012702  057464                  MOV     #TS32D4,R2              ;
```

```
11832  057334  004767  000046                JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
11833  057340  020327  000010                CMP     R3,#10                  ;CHECK FPS
11834  057344  001401                        BEQ     2$                      ;OK GO ON
11835  057346  104003                        ERROR   .3                  ;FPP ERROR
11836                                                                      ;NO GO TO ERROR
11837  057350  170767  123566          2$:   NEGF    TSTLOC+20                       ; TEST INSTRUCTION
11838  057354  170203                        STFPS   R3                      ;GET FPS
11839  057356  012705  003142                MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
11840  057362  012702  057474                MOV     #TS32D5,R2              ;
11841  057366  004767  000014                JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
11842  057372  020327  000004                CMP     R3,#4                   ;CHECK FPS
11843  057376  001401                        BEQ     3$                      ;OK GO ON
11844  057400  104003                        ERROR   .3                  ;FPP ERROR
11845                                                                      ;NO GO TO ERROR
11846  057402                          3$:
11847                                  ;
11848  057402  000167  000076                JMP     FIN32
11849                                  ;
11850  057406  012701  000004          CHEC32: MOV     #4,R1                   ;INIT COUNTER
11851  057412  022522                  1$:   CMP     (R5)+,(R2)+             ;IS DATA CORRECT
11852  057414  001401                        BEQ     2$                      ;YES GO ON
11853  057416  104003                        ERROR   .3                  ;FPP ERROR
11854                                                                      ;NO GO TO ERROR
11855  057420  077104                  2$:   SOB     R1,1$                   ;ARE WE DONE
11856  057422  000207                        RTS     PC                      ;RETURN
11857                                  ;
11858  057424  170000                TS32D0: .WORD   170000
11859  057426  003541                        .WORD   3541
11860  057430  177777                        .WORD   177777
11861  057432  172710                        .WORD   172710
11862  057434  070000                TS32D1: .WORD   70000
11863  057436  003541                        .WORD   3541
11864  057440  177777                        .WORD   177777
11865  057442  172710                        .WORD   172710
11866  057444  000177                        .WORD   177
11867  057446  100000                        .WORD   100000
11868  057450  177777                        .WORD   177777
11869  057452  177007                        .WORD   177007
11870  057454  070000                TS32D3: .WORD   70000
11871  057456  003541                        .WORD   3541
11872  057460  177777                        .WORD   177777
11873  057462  172710                        .WORD   172710
11874  057464  170000                TS32D4: .WORD   170000
11875  057466  003541                        .WORD   3541
11876  057470  177777                        .WORD   177777
11877  057472  172710                        .WORD   172710
11878  057474  000000                TS32D5: .WORD   0
11879  057476  000000                        .WORD   0
11880  057500  177777                        .WORD   177777
11881  057502  177007                        .WORD   177007
11882  057504  000240                FIN32:  NOP
11883                                  ;
11884                                  ;
11885                                  ;----- -------------------------
11886                                  ;TEST  NEGD
11887                                  ;
```

```
11888                                           ;
11889  057506                                   TSFP33:
11890                                           ;
11891  057506  005037  132646                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11892  057512  012705  000200                  MOV     #200,R5                 ;SETUP TO LOAD FPS
11893  057516  170105                          LDFPS   R5                      ;SET FD=1
11894  057520  012701  000014                  MOV     #12.,R1                 ;INIT COUNTER
11895  057524  012704  003122                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11896  057530  012703  057700                  MOV     #TS33D0,R3              ;SETUP POINTER TO TEST VALUE
11897  057534  012324                  100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11898  057536  077102                          SOB     R1,100$                 ;ARE WE DONE
11899  057540  170767  123356                  NEGD    TSTLOC                  ; TEST INSTRUCTION
11900  057544  170203                          STFPS   R3                      ;GET FPS
11901  057546  012705  003122                  MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
11902  057552  012702  057730                  MOV     #TS33D3,R2              ;
11903  057556  004767  000100                  JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
11904  057562  020327  000200                  CMP     R3,#200                 ;CHECK FPS
11905  057566  001401                          BEQ     1$                      ;OK GO ON
11906  057570  104003                          ERROR   +3              ;FPP ERROR
11907                                           ;NO GO TO ERROR
11908  057572  170767  123334          1$:     NEGD    TSTLOC+10                       ; TEST INSTRUCTION
11909  057576  170203                          STFPS   R3                      ;GET FPS
11910  057600  012705  003132                  MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
11911  057604  012702  057740                  MOV     #TS33D4,R2              ;
11912  057610  004767  000046                  JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
11913  057614  020327  000210                  CMP     R3,#210                 ;CHECK FPS
11914  057620  001401                          BEQ     2$                      ;OK GO ON
11915  057622  104003                          ERROR   +3              ;FPP ERROR
11916                                           ;NO GO TO ERROR
11917  057624  170767  123312          2$:     NEGD    TSTLOC+20                       ; TEST INSTRUCTION
11918  057630  170203                          STFPS   R3                      ;GET FPS
11919  057632  012705  003142                  MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
11920  057636  012702  057750                  MOV     #TS33D5,R2              ;
11921  057642  004767  000014                  JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
11922  057646  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11923  057652  001401                          BEQ     3$                      ;OK GO ON
11924  057654  104003                          ERROR   +3              ;FPP ERROR
11925                                           ;NO GO TO ERROR
11926  057656                           3$:
11927                                           ;
11928  057656  000167  000076                  JMP     FIN33
11929                                           ;
11930  057662  012701  000004          CHEC33: MOV     #4,R1                   ;INIT COUNTER
11931  057666  022522                  1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
11932  057670  001401                          BEQ     2$                      ;YES GO ON
11933  057672  104003                          ERROR   +3              ;FPP ERROR
11934                                           ;NO GO TO ERROR
11935  057674  077104                  2$:     SOB     R1,1$                   ;ARE WE DONE
11936  057676  000207                          RTS     PC                      ;RETURN
11937                                           ;
11938  057700  170000                  TS33D0: .WORD   170000
11939  057702  003541                          .WORD   3541
11940  057704  177777                          .WORD   177777
11941  057706  172710                          .WORD   172710
11942  057710  070000                  TS33D1: .WORD   70000
11943  057712  003541                          .WORD   3541
```

```
11944  057714  177777                          .WORD   177777
11945  057716  172710                          .WORD   172710
11946  057720  000177                  TS33D2: .WORD   177
11947  057722  100000                          .WORD   100000
11948  057724  177777                          .WORD   177777
11949  057726  177007                          .WORD   177007
11950  057730  070000                  TS33D3: .WORD   70000
11951  057732  003541                          .WORD   3541
11952  057734  177777                          .WORD   177777
11953  057736  172710                          .WORD   172710
11954  057740  170000                  TS33D4: .WORD   170000
11955  057742  003541                          .WORD   3541
11956  057744  177777                          .WORD   177777
11957  057746  172710                          .WORD   172710
11958  057750  000000                  TS33D5: .WORD   0
11959  057752  000000                          .WORD   0
11960  057754  000000                          .WORD   0
11961  057756  000000                          .WORD   0
11962  057760  000240                  FIN33:  NOP
11963                                  ;
11964                                  ;
11965                                  ;
11966                                  ;---------------------------------
11967                                  ;TEST LDD MODE 0, ILLEGAL AC7
11968                                  ;
11969                                          ;
11970                                          ;
11971                                          ;
11972  057762                          MFSRCM0:
11973
11974                                  ;
11975  057762  012704  047600                  MOV     #47600,R4               ;SETUP FPP STATUS
11976  057766  170104                          LDFPS   R4                      ;LOAD FP STATUS
11977  057770  012702  003062                  MOV     #RECFEC,R2              ;POINT TO RECEIVED FEC MEMORY
11978  057774  172407                  1$:     LDD     R7,AC0                  ;*TEST INSTRUCTION
11979                                                                           ;LOAD AC0 FROM ILLEGAL AC7
11980  057776  170201                          STFPS   R1                      ;SAVE FPP STATUS
11981  060000  022701  147600                  CMP     #147600,R1              ;VERIFY FER BIT SET
11982  060004  001401                          BEQ     2$                      ;BRANCH IF GOOD ERROR CONDITION
11983  060006  104003                          ERROR   ;3              ;FPP ERROR
11984                                          ;THE FER BIT DIDNT SET
11985  060010  170312                  2$:     STST    (R2)                    ;SAVE FEC AND FEA
11986  060012  022722  000002                  CMP     #2,(R2)+                ;VERIFY FEC CONTENTS
11987  060016  001401                          BEQ     3$                      ;BRANCH IF GOOD
11988  060020  104003                          ERROR   ;3              ;FPP ERROR
11989                                          ;FEC NE 2 (OPCODE ERROR)
11990  060022  022722  057774          3$:     CMP     #1$,(R2)+               ;VERFIY FEA CONTENTS
11991  060026  001401                          BEQ     4$                      ;BRANCH FI GOOD
11992  060030  104003                          ERROR   ;3              ;FPP ERROR
11993                                          ;FEA NOT CORRECT ERROR ADDRESS
11994  060032                          4$:
11995
11996
11997                                  ;
11998                                  ;
11999                                  ;---------------------------------
```

```
12000                                   ;TEST LDD MODE2
12001                                   ;
12002                                           ;
12003                                           ;
12004                                           ;
12005  060032                           MLDDM2:
12006
12007                                   ;
12008  060032  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12009  060036  012704  003174                  MOV     #TAB1,R4                ;POINT TO GOOD DATA
12010  060042  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12011  060046  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12012  060050  172424                          LDD     (R4)+,AC0               ;*TEST INSTRUCTION - MODE 2
12013  060052  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12014  060054  174021                          STD     AC0,(R1)+               ;SAVE TEST RESULT MODE 2
12015  060056  020203                          CMP     R2,R3                   ;VERIFY FPP STATUS
12016  060060  001401                          BEQ     1$                      ;BRANCH IF GOOD
12017  060062  104003                          ERROR   +3              ;FPP ERROR
12018                                          ;BAD FPP STSTUS
12019  060064  022704  003204          1$:     CMP     #TAB1+10,R4
12020  060070  001401                          BEQ     2$                              ;VERFIY AUTO-INCR
12021  060072  104003                          ERROR   +3                      ;BRANCH IF GOOD
12022                                  ;FPP ERROR
12023  060074  012704  003174          2$:     MOV     #TAB1,R4                ;BAD AUTO-INCR
12024  060100  162701  000010                  SUB     #10,R1                          ;POINT TO RECEIVED DATA
12025  060104  004767  052562                  JSR     R7,DATVER               ;RETURN R1 TO PROPER VALUE
12026  060110  005767  122744                  TST     COUNT                   ;VERFIY DATA FROM FPP
12027  060114  001401                          BEQ     3$                      ;SEE IF COUNTER=0
12028  060116  104003                          ERROR   +3                      ;BRANCH IF GOOD COMPARE
12029                                  ;FPP ERROR
12030  060120                           3$:    ;BAD DATA FROM FPP
12031
12032                                   ;
12033                                   ;
12034                                   ;
12035                                   ;
12036                                   ;---------------------------------
12037                                   ;TEST LDD MODE 3
12038                                   ;
12039                                           ;
12040                                           ;
12041                                           ;
12042  060120                           MLDDM3:
12043
12044                                   ;
12045  060120  012737  003102  003124          MOV     #RECDST,@#TSTLOC+2              ;POINT TO RECEIVED DATA LOCATION
12046  060126  012701  003124                  MOV     #TSTLOC+2,R1            ;SETUP STD IN MODE 3
12047  060132  012737  003204  003122          MOV     #TAB2,@#TSTLOC          ;POINT TO DATA TABLE
12048  060140  012704  003122                  MOV     #TSTLOC,R4                      ;POINT TO GOOD DATA
12049  060144  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12050  060150  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12051  060152  172434                          LDD     @(R4)+,AC0                      ;*TEST INSTRUCTION - MODE 2
12052  060154  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12053  060156  174031                          STD     AC0,@(R1)+                      ;SAVE TEST RESULT IN MODE 3
12054  060160  022703  047740                  CMP     #47740,R3                       ;VERIFY FPP STATUS
12055  060164  001401                          BEQ     1$                      ;BRANCH IF GOOD
```

M1


```
12056  060166  104003                          ERROR   ;3              ;FPP ERROR
12057                                                                  ;BAD FPP STSTUS
12058  060170  022704  003124          1$:     CMP     @TSTLOC+2,R4                    ;VERFIY AUTO-INCR
12059  060174  001401                          BEQ     2$              ;BAD AUTO-DEC ON LDD
12060  060176  104003                          ERROR   ;3              ;FPP ERROR
12061                                                                  ;BAD AUTO-INC
12062  060200  022701  003126          2$:     CMP     @TSTLOC+4,R1                    ; TEST STD AUTO-INC
12063  060204  001401                          BEQ     3$              ;BRANCH IF GOOD
12064  060206  104003                          ERROR   ;3              ;FPP ERROR
12065                                                                  ;BAD AUTO-INCR
12066  060210  012704  003204          3$:     MOV     @TAB2,R4        ;POINT TO RECEIVED DATA
12067  060214  012701  003102                  MOV     @RECDST,R1      ;POINT TO RECEIVED DATA
12068  060220  004767  052446                  JSR     R7,DATVER       ;VERFIY DATA FROM FPP
12069  060224  005767  122630                  TST     COUNT           ;SEE IF COUNTER=0
12070  060230  001401                          BEQ     4$              ;BRANCH IF GOOD COMPARE
12071  060232  104003                          ERROR   ;3              ;FPP ERROR
12072                                                                  ;BAD DATA FROM FPP
12073  060234                           4$:
12074
12075                                    ;
12076                                    ;
12077                                    ;
12078                                    ;
12079                                    ;--------------------------------
12080                                    ;TEST LDF, STD MODE 4
12081                                    ;
12082                                            ;
12083                                            ;
12084                                            ;
12085  060234                           MLDDM4:
12086
12087                                    ;
12088  060234  012701  003106                  MOV     @RECDST+4,R1    ;POINT TO RECEIVED DATA LOCATION
12089  060240  012704  003220                  MOV     @TAB3+4,R4      ;POINT TO GOOD DATA
12090  060244  012705  003254                  MOV     @TAB6,R5        ;CLEAR OUT AC0
12091  060250  170127  000200                  LDFPS   @200            ;SET TO DOUBLE
12092  060254  17'415                          LDD     (R5),AC0                ;AC0=0
12093  060256  0 .702  047550                  MOV     @47550,R2       ;LOAD GOOD STATUS FLOATING
12094  060262  1.0102                          LDFPS   R2              ;LOAD FPP STATUS - DOUBLE,ID
12095  060264  172444                          LDF     -(R4),AC0               ;*TEST INSTRUCTION - MODE 4
12096  060266  170203                          STFPS   R3              ;SAVE TEST FPP STATUS
12097  060270  012702  047750                  MOV     @47750,R2       ;SET TO DOUBLE MODE
12098  060274  170102                          LDFPS   R2              ;SET FPP TO DOUBLE
12099  060276  174041                          STD     AC0,-(R1)               ;SAVE TEST RESULT
12100  060300  022703  047540                  CMP     @47540,R3               ;VERIFY FPP STATUS
12101  060304  001401                          BEQ     1$              ;BRANCH IF GOOD
12102  060306  104003                          ERROR   ;3      ;FPP ERROR
12103                                                          ;BAD FPP STSTUS
12104  060310  022704  003214          1$:     CMP     @TAB3,R4                ;VERFIY AUTO-DEC
12105  060314  001401                          BEQ     2$              ;BRANCH IF GOOD
12106  060316  104003                          ERROR   ;3      ;FPP ERROR
12107                                                          ;BAD AUTO-INCR
12108  060320  012704  003214          2$:     MOV     @TAB3,R4                ;POINT TO RECEIVED DATA
12109  0 0324  004767  052342                  JSR     R7,DATVER       ;VERFIY DATA FROM FPP
12110  0 0330  005767  122524                  TST     COUNT           ;SEE IF COUNTER=0
12111  060334  001401                          BEQ     3$              ;BRANCH IF GOOD COMPARE
```

```
12112  060336  104003                          ERROR   ;3              ;FPP ERROR
12113                                                                  ;BAD DATA FROM FPP
12114  060340                          3$:
12115
12116                                  ;
12117                                  ;
12118                                  ;
12119                                  ;
12120                                  ;---------------------------------
12121                                  ;TEST LDD MODE 5
12122                                  ;
12123                                          ;
12124                                          ;
12125                                          ;
12126  060340                          MLDDM5:
12127
12128                                  ;
12129  060340  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12130  060344  012704  003124                  MOV     #TSTLOC+2,R4                    ;POINT TO GOOD DATA
12131  060350  012737  003174  003122          MOV     #TAB1,@#TSTLOC                  ;SET UP MODE 5 POINTER TO DATA
12132  060356  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12133  060362  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12134  060364  172454                          LDD     @-(R4),AC0              ;*TEST INSTRUCTION - MODE 5
12135  060366  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12136  060370  174011                          STD     AC0,(R1)                ;SAVE TEST RESULT
12137  060372  020203                          CMP     R2,R3                   ;VERIFY FPP STATUS
12138  060374  001401                          BEQ     1$                      ;BRANCH IF GOOD
12139  060376  104003                          ERROR   ;3              ;FPP ERROR
12140                                                                  ;BAD FPP STATUS
12141  060400  022704  003122          1$:     CMP     #TSTLOC,R4                      ;VERFIY AUTO-DEC
12142  060404  001401                          BEQ     2$              ;BRANCH IF GOOD
12143  060406  104003                          ERROR   ;3              ;FPP ERROR
12144                                                                  ;BAD AUTO-DEC
12145  060410  012704  003174          2$:     MOV     #TAB1,R4                        ;POINT TO EXPECTED DATA
12146  060414  004767  052252                  JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12147  060420  005767  122434                  TST     COUNT                   ;SEE IF COUNTER=0
12148  060424  001401                          BEQ     3$                      ;BRANCH IF GOOD COMPARE
12149  060426  104003                          ERROR   ;3              ;FPP ERROR
12150                                                                  ;BAD DATA FROM FPP
12151  060430                          3$:
12152
12153                                  ;
12154                                  ;
12155                                  ;--------------- ------ -- - -----
12156                                  ;TEST LDD MODE 6
12157                                  ;
12158                                          ;
12159                                          ;
12160                                          ;
12161  060430                          MLDDM6:
12162
12163                                  ;
12164  060430  012701  003302                  MOV     #RECDST+200,R1          ;POINT TO RECEIVED DATA LOCATION
12165  060434  012704  003004                  MOV     #TAB2-200,R4                    ;SETUP R4 FOR MODE 6
12166  060440  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12167  060444  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
```

```
12168  060446  172464  000200                  LDD     200(R4),AC0             ;LDD MODE 6
12169  060452  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12170  060454  174061  177600                  STD     AC0,-200(R1)                    ;SAVE TEST RESULT
12171  060460  022703  047740                  CMP     #47740,R3                       ;VERIFY FPP STATUS
12172  060464  001401                          BEQ     1$              ;BRANCH IF GOOD
12173  060466  104003                          ERROR   .3              ;FPP ERROR
12174                                                                  ;BAD FPP STATUS
12175  060470  162701  000200          1$:     SUB     #200,R1                 ;R1=RECDST
12176  060474  062704  000200          2$:     ADD     #200,R4                 ;POINT TO EXPECTED DATA
12177  060500  004767  052166                  JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12178  060504  005767  122350                  TST     COUNT                   ;SEE IF COUNTER=0
12179  060510  001401                          BEQ     3$                      ;BRANCH IF GOOD COMPARE
12180  060512  104003                          ERROR   .3              ;FPP ERROR
12181                                                                  ;BAD DATA FROM FPP
12182  060514                          3$:
12183
12184                                  ;
12185                                  ;
12186                                  ;--------  --   ----                   -
12187                                  ;TEST LDD MODE 7
12188                                  ;
12189                                          ;
12190                                          ;
12191                                          ;
12192  060514                          MLDDM7:
12193
12194                                  ;
12195  060514  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12196  060520  005004                          CLR     R4                      ;R4=0
12197  060522  012727  003174  003122          MOV     #TAB1,#TSTLOC           ;POINTER FOR MODE 7 GOOD DATA
12198  060530  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12199  060534  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12200  060536  172474  003122                  LDD     @TSTLOC(R4),AC0         ;*TEST INSTRUCTION - MODE 7
12201  060542  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12202  060544  174011                          STD     AC0,(R1)                        ;SAVE TEST RESULT
12203  060546  020203                          CMP     R2,R3                   ;VERIFY FPP STATUS
12204  060550  001401                          BEQ     1$                      ;BRANCH IF GOOD
12205  060552  104003                          ERROR   .3              ;FPP ERROR
12206                                                                  ;BAD FPP STATUS
12207  060554  005704                  1$:     TST     R4                              ;VERFIY CONTENTS OF R4
12208  060556  001401                          BEQ     2$                              ;BRANCH IF GOOD
12209  060560  104003                          ERROR   .3              ;FPP ERROR
12210                                                                  ;BAD R4
12211  060562  012704  003174          2$:     MOV     #TAB1,R4                                ;POINT TO RECEIVED DATA
12212  060566  004767  052100                  JSR     R7,DATVER                       ;VERFIY DATA FROM FPP
12213  060572  005767  122262                  TST     COUNT                           ;SEE IF COUNTER=0
12214  060576  001401                          BEQ     3$                              ;BRANCH IF GOOD COMPARE
12215  060600  104003                          ERROR   .3              ;FPP ERROR
12216                                                                  ;BAD DATA FROM FPP
12217  060602                          3$:
12218
12219                                  ;
12220                                  ;
12221                                  ;----- ---------------------------
12222                                  ;TEST LDD MODE 27 - ONLY 16 BITS ARE LOADED OR STORED
12223                                  ;
```

```
12224                                           ;
12225                                           ;
12226                                           ;
12227  060602                           MLDM27:
12228
12229                                           ;
12230  060602  012701  003102                   MOV     #RECDST,R1          ;POINT TO RECEIVED DATA LOCATION
12231  060606  012704  003234                   MOV     #TAB5,R4                    ;POINT TO GOOD DATA
12232  060612  012702  047750                   MOV     #47750,R2           ;LOAD GOOD STATUS
12233  060616  005005                           CLR     R5                  ;R5=0
12234  060620  170102                           LDFPS   R2                  ;LOAD FPP STATUS - DOUBLE,ID
12235  060622  172427  043243                   LDD     #5205,AC0           ;*TEST INSTRUCTION - MODE 27
12236  060626  005205                           INC     R5
12237  060630  005205                           INC     R5
12238  060632  005205                           INC     R5                  ; TEST PROPER PC PATH
12239  060634  022705  000003                   CMP     #3,R5               ;VERIFY ONLY 3 PC INCREMENT
12240  060640  001401                           BEQ     1$                  ;BRANCH IF PROPER PC ACTION
12241  060642  104003                           ERROR   .3                ;FPP ERROR
12242                                           ;BAD MODE 27 LOAD
12243  060644  170203                   1$:     STFPS   R3                  ;SAVE TEST FPP STATUS
12244  060646  174011                           STD     AC0,(R1)                    ;SAVE TEST RESULT
12245  060650  022703  047740                   CMP     #47740,R3                   ;VERIFY FPP STATUS
12246  060654  001401                           BEQ     2$                  ;BRANCH IF GOOD
12247  060656  104003                           ERROR   .3                ;FPP ERROR
12248                                           ;BAD FPP STATUS
12249  060660  004767  052006           2$:     JSR     R7,DATVER           ;VERFIY DATA FROM FPP
12250  060664  005767  122170                   TST     COUNT               ;SEE IF COUNTER=0
12251  060670  001401                           BEQ     3$                  ;BRANCH IF GOOD COMPARE
12252  060672  104003                           ERROR   .3                ;FPP ERROR
12253                                           ;BAD DATA FROM FPP
12254  060674                           3$:
12255
12256                                           ;
12257                                           ;
12258                                           ;-------------- ---   -- --------
12259                                           ;TEST ADDF, ADDD, SUBF, SUBD   AC0=0 FSRC=0;
12260                                                   ;
12261                                                   ;
12262                                                   ;
12263  060674                           MNNRM1:
12264
12265                                           ;
12266  060674  012704  003254                   MOV     #TAB6,R4                    ;POINT TO FSRC TEST DATA
12267  060700  005067  122202                   CLR     RECDST+4                    ;CLEAR OUT RECEIVED DATA TABLE
12268  060704  005067  122200                   CLR     RECDST+6                    ;
12269  060710  012702  040000                   MOV     #40000,R2                   ;SET UP GOOD STATUS
12270  060714  170102                           LDFPS   R2                          ;LOAD FPP STATUS, FLOATING
12271  060716  172414                           LDF     (R4),AC0                    ;LOAD AC0 WITH 0
12272  060720  172014                           ADDF    (R4),AC0                    ;0+0
12273  060722  170203                           STFPS   R3                          ;SAVE STATUS
12274  060724  022703  040004                   CMP     #40004,R3                   ;VERIFY STATUS
12275  060730  001401                           BEQ     1$                          ;BRANCH IF GOOD
12276  060732  104003                           ERROR   .3                        ;FPP ERROR
12277                                           ;BAD FPP STATUS
12278  060734  012701  003102           1$:     MOV     #RECDST,R1                  ;POINT TO RECEIVERD DATA
12279  060740  174011                           STF     AC0,(R1)                    ;SAVE DATA
```

D2


```
12280  060742  004767  051724                  JSR     R7,DATVER               ;VERIFY DATA
12281  060746  005767  122106                  TST     COUNT                   ;
12282  060752  001401                          BEQ     2$                      ;BRANCH IF GOOD
12283  060754  104003                          ERROR   .3                  ;FPP ERROR
12284                                                                       ;BAD DATA IN ACO
12285  060756  012702  040200          2$:     MOV     #40200,R2               ;LOAD FLOATING STATUS
12286  060762  170102                          LDFPS   R2                      ;
12287  060764  172414                          LDD     (R4),AC0                ;LOAD AC0 WITH 0
12288  060766  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12289  060770  174011                          STD     AC0,(R1)                ;SAVE DATA
12290  060772  170203                          STFPS   R3                      ;SAVE FPS
12291  060774  022703  040204                  CMP     #40204,R3               ;VERFIY STATUS
12292  061000  001401                          BEQ     3$                      ;BRANCH IF GOOD
12293  061002  104003                          ERROR   .3                  ;FPP ERROR
12294                                                                       ;BAD FPS
12295  061004  004767  051662          3$:     JSR     R7,DATVER                       ;VERFIY DATA
12296  061010  005737  003060                  TST     @#COUNT                 ;VERIFY RESULT
12297  061014  001401                          BEQ     44$                     ;BRANCH IF GOOD
12298  061016  104003                          ERROR   .3                  ;FPP ERROR
12299                                                                       ;BAD AC0
12300  061020  172414          44$:    LDD     (R4),AC0                ;SETUP DATA
12301  061022  173014                  SUBD    (R4),AC0                ;*TEST INSTRUCTION
12302  061024  170203                  STFPS   R3                      ;SAVE STATUS
12303  061026  022703  040204                  CMP     #40204,R3               ;VERFIY STATUS
12304  061032  001401                          BEQ     4$                      ;BRANCH IF GOOD
12305  061034  104003                          ERROR   .3                  ;FPP ERROR
12306                                                                       ;BAD FPS
12307  061036  174011          4$:     STD     AC0,(R1)                ;SAVE AC0 DATA
12308  061040  004767  051626                  JSR     R7,DATVER                       ;VERFIY DATA
12309  061044  005767  122010                  TST     COUNT                   ;
12310  061050  001401                          BEQ     5$                      ;BRANCH IF GOOD
12311  061052  104003                          ERROR   .3                  ;FPP ERROR
12312                                                                       ;BAD AC0
12313  061054  170127  000000          5$:     LDFPS   #0                      ;STORE FPP STATUS
12314  061060  172414                          LDD     (R4),AC0                ;LOAD AC0
12315  061062  173014                          SUBF    (R4),AC0                        ;0-0
12316  061064  170203                          STFPS   R3                      ;SAVE STATUS
12317  061066  174011                          STD     AC0,(R1)                ;SAVE AC0
12318  061070  022703  000004                  CMP     #4,R3                   ;VERFIY STATUS
12319  061074  001401                          BEQ     6$                      ;BRANCH IF GOOD
12320  061076  104003                          ERROR   .3                  ;FPP ERROR
12321                                                                       ;BAD FPS
12322  061100  004767  051566          6$:     JSR     R7,DATVER                       ;VERIFY DATAT
12323  061104  005767  121750                  TST COUNT                       ;
12324  061110  001401                          BEQ     7$                  ;BRANC IF GOOD
12325  061112  104003                          ERROR   .3                  ;FPP ERROR
12326                                                                       ;BAD AC0
12327  061114                          7$:
12328
12329
12330                                   ;
12331                                   ;
12332                                   ;---------------------------------
12333                                   ;TEST ADDF,SUBD - FSRC=0, AC0 NE 0
12334                                   ;
12335                                           ;
```

E2


```
12336                                           ;
12337                                           ;
12338  061114                           MNNRM2:
12339
12340                                           ;
12341  061114  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12342  061120  012705  003254                  MOV     #TAB6,R5                ;POINT TO SOURCE DATA TABLE
12343  061124  012704  003204                  MOV     #TAB2,R4                ;POINT TO ACO DATA
12344  061130  170127  000200                  LDFPS   #200                    ;SET TO DOUBLE FOR CLEAR
12345  061134  172415                          LDD     (R5),AC0                ;
12346  061136  005002                          CLR     R2                      ;SETUP FPP STATUS
12347  061140  170102                          LDFPS   R2                      ;LOAD FPS
12348  061142  172414                          LDF     (R4),AC0                ;LOAD AC0
12349  061144  172015                          ADDF    (R5),AC0                        ;*TEST INSTRUCTION
12350  061146  170203                          STFPS   R3                      ;SAVE STATUS
12351  061150  174011                          STF     AC0,(R1)                ;SAVE AC0
12352  061152  022703  000000                  CMP     #0,R3                   ;VERFIY NEGATIVE RESULT
12353  061156  001401                          BEQ     1$                      ;BRANCH IF GOOD
12354  061160  104003                          ERROR   .3              ;FPP ERROR
12355                                           ;BAD FPS
12356  061162  012704  003224          1$:     MOV     #TAB4,R4                ;POINT TO EXPECTED DATA
12357  061166  0C4767  051500                  JSR     R7,DATVER                       ;VERFIY AC0
12358  061172  005767  121662                  TST     COUNT                   ;CHECK RESULT
12359  061176  001401                          BEQ     2$                      ;BRANCH IF GOOD
12360  061200  104003                          ERROR   .3              ;FPP ERROR
12361                                           ;BAD AC0
12362  061202  170127  000200          2$:     LDFPS   #200                    ;SET STATUS TO DUOBLE NODE
12363  061206  172414                          LDD     (R4),AC0                ;LOAD AC0 WITH A VALUE
12364  061210  173015                          SUBD    (R5),AC0                ;*TEST INSTRUCTION
12365  061212  170203                          STFPS   R3                      ;SAVE FPP STATUS
12366  061214  174011                          STD     AC0,(R1)                ;SAVE AC0
12367  061216  022703  000200                  CMP     #200,R3         ;VERIFY RESULT
12368  061222  001401                          BEQ     3$                      ;BRANCH IF GOOD
12369  061224  104003                          ERROR   .3              ;FPP ERROR
12370                                           ;BAD SUBD
12371  061226  012704  003224          3$:     MOV     #TAB4,R4                ;POINT TO EXPECTED
12372  061232  004767  051434                  JSR     R7,DATVER               ;VERIFY AC0
12373  061236  005767  121616                  TST     COUNT                   ;
12374  061242  001401                          BEQ     4$                      ;BRANCH IF GOOD AC0
12375  061244  104003                          ERROR   .3              ;FPP ERROR
12376                                           ;BAD AC0
12377  061246                           4$:
12378
12379
12380                                           ;
12381                                           ;
12382                                           ;---------------------------------
12383                                           ;TEST ADDD, SUBF - FSRC NE 0, AC0=0
12384                                           ;
12385                                                   ;
12386                                                   ;
12387                                                   ;
12388  061246                           MNNRM3:
12389
12390                                           ;
12391  061246  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
```

F2


```
12392  061252  012705  003254          MOV     #TAB6,R5        ;POINT TO ACO DATA TABLE
12393  061256  012704  003174          MOV     #TAB1,R4        ;POINT TO FSRC DATA
12394  061262  012702  000200          MOV     #200,R2         ;SETUP FPP STATUS
12395  061266  170102                  LDFPS   R2              ;LOAD FPS
12396  061270  172415                  LDD     (R5),AC0        ;LOAD AC0
12397  061272  172014                  ADDD    (R4),AC0                ;*TEST INSTRUCTION
12398  061274  170203                  STFPS   R3              ;SAVE STATUS
12399  061276  174011                  STD     AC0,(R1)        ;SAVE AC0
12400  061300  022703  000210          CMP     #210,R3         ;VERFIY NEGATIVE RESULT
12401  061304  001401                  BEQ     1$              ;BRANCH IF GOOD
12402  061306  104003                  ERROR   ;3              ;FPP ERROR
12403                                                          ;BAD FPS
12404  061310  004767  051356  1$:     JSR     R7,DATVER               ;VERFIY AC0
12405  061314  005767  121540          TST     COUNT           ;CHECK RESULT
12406  061320  001401                  BEQ     2$              ;BRANCH IF GOOD
12407  061322  104003                  ERROR   ;3              ;FPP ERROR
12408                                                          ;BAD AC0
12409  061324  170127  000200  2$:     LDFPS   #200            ;SET STATUS TO DUOBLE NODE
12410  061330  172415                  LDF     (R5),AC0        ;LOAD AC0 WITH A VALUE
12411  061332  173014                  SUBD    (R4),AC0        ;*TEST INSTRUCTION
12412  061334  170203                  STFPS   R3              ;SAVE FPP STATUS
12413  061336  174011                  STF     AC0,(R1)                ;SAVE AC0
12414  061340  022703  000200          CMP     #200,R3         ;VERIFY RESULT
12415  061344  001401                  BEQ     3$              ;BRANCH IF GOOD
12416  061346  104003                  ERROR   ;3              ;FPP ERROR
12417                                                          ;BAD SUBD
12418  061350  012704  003444  3$:     MOV     #TAB18,R4       ;POINT TO EXPECTED DATA
12419  061354  004767  051312          JSR     R7,DATVER       ;VERIFY AC0
12420  061360  005767  121474          TST     COUNT           ;
12421  061364  001401                  BEQ     4$              ;BRANCH IF GOOD AC0
12422  061366  104003                  ERROR   ;3              ;FPP ERROR
12423                                                          ;BAD AC0
12424  061370                  4$:
12425
12426
12427                          ;
12428                          ;
12429                          ;---------------------------------
12430                          ;TEST  ADDF, SUBD - EXP(AC0) = EXP(FSRC)
12431                          ;
12432                                  ;
12433                                  ;
12434                                  ;
12435  061370                  MNNRM4:
12436
12437                          ;
12438  061370  012702  003240          MOV     #3240,R2                ;SET FIU,FD,FT
12439  061374  170102                  LDFPS   R2
12440  061376  012704  003274          MOV     #TAB7,R4                        ;SET FSRC
12441  061402  012705  003304          MOV     #TAB8,R5        ;SETUP AC0
12442  061406  012701  003102          MOV     #RECDST,R1      ;POINT TO RECEIVED DATA
12443  061412  172415                  LDD     (R5),AC0                ;LOAD AC0
12444  061414  172014                  ADDD    (R4),AC0        ;*TEST INSTRUCTION
12445  061416  174011                  STD     AC0,(R1)        ;SAVE TEST RESULT
12446  061420  012704  003314          MOV     #TAB9,R4        ;POINT TO EXPECTED DATA
12447  061424  004767  051242          JSR     R7,DATVER               ;VERIFY AC0 DATA
```

```
12448  061430  005767  121424                  TST     COUNT                   ;
12449  061434  001401                          BEQ     1$                      ;BRANCH IF GOOD
12450  061436  104003                          ERROR   .3              ;FPP ERROR
12451                                                                   ;BAD ADDD
12452  061440  012704  003304          1$:     MOV     #TAB8,R4                ;
12453  061444  012703  003304                  MOV     #TAB8,R3                ;SETUP SAME ACO
12454  061450  012702  003200                  MOV     #3200,R2                ;ROUND MODE
12455  061454  170102                          LDFPS   R2
12456  061456  172413                          LDD     (R3),AC0                ;LOAD AC0
12457  061460  061400                          ADD     (R4),AC0                ;*TEST INSTRUCTION
12458  061462  174011                          STD     AC0,(R1)                ;SAVE DATA
12459  061464  004767  051202                  JSR     R7,DATVER               ;VERIFY AC0
12460  061470  005767  121364                  TST     COUNT
12461  061474  001401                          BEQ     2$                      ;BRANCH IF GOOD
12462  061476  104003                          ERROR   .3              ;FPP ERROR
12463                                                                   ;BAD ROUND RESULT
12464  061500                          2$:
12465
12466
12467                                  ;
12468                                  ;
12469                                  ;---------------------------------
12470                                  ;TEST ADD - EXP(FSRC) > EXP(AC0)
12471                                  ;
12472                                          ;
12473                                          ;
12474                                          ;
12475  061500                          MXDF1:
12476
12477                                  ;
12478  061500  012702  003200                  MOV     #3200,R2                ;R2=FPP STATUS
12479  061504  170102                          LDFPS   R2                      ;LOAD FPS STATUS
12480  061506  012704  003334                  MOV     #TAB11,R4               ;POINT TO FSRC DATA
12481  061512  012701  003102                  MOV     #RECDST,R1              ;POINT TO AC0 RESULT
12482  061516  012705  003324                  MOV     #TAB10,R5               ;POINT TO AC0 DATA
12483  061522  172415                          LDD     (R5),AC0                ;LOAD AC0 DATA
12484  061524  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTIONS
12485  061526  170203                          STFPS   R3                      ;SAVE FPP STATUS
12486  061530  174011                          STD     AC0,(R1)                ;SAVE AC0 DATA
12487  061532  022703  003200                  CMP     #3200,R3                ;VERIFY FPP STATUS
12488  061536  001401                          BEQ     1$                      ;BRANCH IF GOOD
12489  061540  104003                          ERROR   .3              ;FPP ERROR
12490                                                                   ;BAD FPP STATUS
12491  061542  012704  003344          1$:     MOV     #TAB11A,R4              ;POINT TO EXPECTED DATA
12492  061546  004767  051120                  JSR     R7,DATVER               ;VERIFY CONTENTS OF AC0
12493  061552  005767  121302                  TST     COUNT                   ;
12494  061556  001401                          BEQ     2$                      ;BRANCH IF GOOD AC0
12495  061560  104003                          ERROR   .3              ;FPP ERROR
12496                                                                   ;BAD AC0, SHOULD = FSRC
12497  061562  012704  003354          2$:     MOV     #TAB12,R4               ;POINT TO FSRC DATA
12498  061566  172415                          LDD     (R5),AC0                ;AC0
12499  061570  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12500  061572  012704  003374                  MOV     #TAB13B,R4              ;POINT TO EXPECTED RESULT
12501  061576  174011                          STD     AC0,(R1)                ;SAVE AC0 DATA INTO RECDAT
12502  061600  004767  051066                  JSR     R7,DATVER               ;VERIFY DATA
12503  061604  005767  121250                  TST     COUNT
```

H2


```
12504  061610  001401                        BEQ     3$                      ;BRANCH IF GOOD DATA
12505  061612  104003                        ERROR   .3                  ;FPP ERROR
12506                                                                     ;BAD ACO DATA
12507  061614  012702  003000          3$:   MOV     #3000,R2                ;GET FPP STATUS DATA
12508  061620  012704  003404                MOV     #TAB14,R4               ;POINT TO FSRC DATA
12509  061624  012705  003414                MOV     #TAB15,R5               ;POINT TO ACO DATA
12510  061630  172415                        LDD     (R5),AC0                ;LOAD AC0
12511  061632  170102                        LDFPS   R2                      ;FPP STATUS = FLOAT, INTERRUPTS ENABLE
12512  061634  172014                        ADDF    (R4),AC0                ;*TEST INSTRUCTION
12513  061636  170127  000200                LDFPS   #200                    ;RESET TO DOUBLE
12514  061642  174011                        STD     AC0, (R1)               ;RECDST=AC0
12515  061644  012704  003324                MOV     #TAB10,R4               ;POINT TO GOOD DATA
12516  061650  004767  051016                JSR     R7,DATVER                       ;VERFIY CONTENTS OF AC0
12517  061654  005767  121200                TST     COUNT                   ;
12518  061660  001401                        BEQ     4$                      ;BRANCH IF GOOD
12519  061662  104003                        ERROR   .3                  ;FPP ERROR
12520                                                                     ;BAD FLOATING ADD
12521  061664  012705  003424          4$:   MOV     #TAB16,R5               ;POINT TO AC0 DATA
12522  061670  170102                        LDFPS   R2                      ;FPP STATUS = FLOAT
12523  061672  172415                        LDF     (R5),AC0                ;LOAD AC0
12524  061674  172014                        ADDF    (R4),AC0                ;*TEST INSTRUCTION
12525  061676  174011                        STD     AC0,(R1)                        ;SAVE AC0  DATA
12526  061700  012704  003434                MOV     #TAB17,R4               ;POINT TO GOOD DATA
12527  061704  004767  050762                JSR     R7,DATVER
12528  061710  005767  121144                TST     COUNT
12529  061714  001401                        BEQ     5$                      ;BRANCH IF GOOD
12530  061716  104003                        ERROR   .3                  ;FPP ERROR
12531                                                                     ;BAD FLOATING ADD
12532  061720                          5$:
12533
12534
12535
12536                                  ;
12537                                  ;
12538                                  ;-----------------------------------
12539                                  ;TEST ADDD WITH NEGATIVE OPERANDS
12540                                  ;
12541                                          ;
12542                                          ;
12543                                          ;
12544  061720                          MNGOP:
12545
12546                                  ;
12547  061720  012702  003200                  MOV     #3200,R2                ;LOAD FPS VALUE
12548  061724  170102                          LDFPS   R2                      ;
12549  061726  012704  003454                  MOV     #TAB21,R4               ;DATA ADDRESS FOR AC0 AND FSR
12550  061732  172414                          LDD     (R4),AC0                ;AC0=100200 0 0 0
12551  061734  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12552  061736  170203                          STFPS   R3                      ;SAVE STATUS
12553  061740  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12554  061744  174011                          STD     AC0,(R1)                ;SAVE AC0 DATA
12555  061746  022703  003210                  CMP     #3210,R3                ;VERIFY STATUS
12556  061752  001401                          BEQ     1$                      ;BRANCH IF GOOD
12557  061754  104003                          ERROR   .3                  ;FPP ERROR
12558                                                                      ;
12559  061756  012704  003464          1$:     MOV     #TAB22,R4               ;POINT TO EXPECTED DATA
```

```
12560  061762  004767  050704                  JSR     R7,DATVER               ;
12561  061766  005767  121066                  TST     COUNT                   ;VERIFY DATA
12562  061772  001401                          BEQ     2$                      ;BRANCH IF GOOD
12563  061774  104003                          ERROR   .3              ;FPP ERROR
12564                                                                  ;
12565                                  ;       !-FSRC! = !ACO!
12566  061776  012704  003454          2$:     MOV     #TAB21,R4               ;POINT TO FSRC DATA
12567  062002  012701  003474                  MOV     #TAB23,R1               ;POINT TO ACO DATA
12568  062006  012737  062024  000244          MOV     #101$,@#FPVEC           ;SETUP FP VECTOR
12569  062014  172411                          LDD     (R1),AC0                ;LOAD AC0
12570  062016  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12571  062020  170000                  100$:   CFCC                            ;COPY FPP CC
12572  062022  104003                          ERROR   .3              ;FPP ERROR
12573                                                                  ;GO TO ERROR
12574  062024  170203                  101$:   STFPS   R3                      ;SAVE FPP STATUS
12575  062026  012701  003102                  MOV     #RECDST,R1                      ;POINT TO RECEIVED DATA TABLE
12576  062032  174011                          STD     AC0,(R1)                ;SAVE AC0 DATA
12577  062034  022703  103200                  CMP     #103200,R3              ;VERIFY STATUS
12578  062040  001401                          BEQ     3$                      ;BRANCH IF GOOD
12579  062042  104003                          ERROR   .3              ;FPP ERROR
12580                                                                  ;BAD STATUS
12581  062044  012605                  3$:     MOV     (SP)+,R5                ;GET ERROR PC
12582  062046  020527  062020                  CMP     R5,#100$                ;VERIFY ERROR ADDRESS ON STACK
12583  062052  001401                          BEQ     102$                    ;BRANCH IF GOOD
12584  062054  104003                          ERROR   .3              ;FPP ERROR
12585                                                                  ;BAD ERROR RETURN ON STACK
12586  062056  005726                  102$:   TST     (SP)+                   ;RESTORE STACK
12587  062060  012704  003504                  MOV     #TAB24,R4               ;POINT TO EXPECTED DATA TABLE
12588  062064  004767  050602                  JSR     R7,DATVER               ;VERIFY DATA
12589  062070  005767  120764                  TST     COUNT
12590  062074  001401                          BEQ     4$                      ;BRANC IF GOOD
12591  062076  104003                          ERROR   .3              ;FPP ERROR
12592                                                                  ;BAD ACO DATA
12593                                  ;       !-AC0! = !FSRC!
12594  062100  012704  003474          4$:     MOV     #TAB23,R4               ;POINT TO FSRC DATA
12595  062104  012701  003454                  MOV     #TAB21,R1               ;POINT TO ACO DATA
12596  062110  012737  062134  000244          MOV     #104$,@#FPVEC           ;SETUP FP VECTOR
12597  062116  012702  003200                  MOV     #3200,R2                ;LOAD FPS VALUE
12598  062122  170102                          LDFPS   R2                      ;
12599  062124  172411                          LDD     (R1),AC0                ;LOAD ACO DATA
12600  062126  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12601  062130  170000                  103$:   CFCC                            ;COPY FPP CC
12602  062132  104003                          ERROR   .3              ;FPP ERROR
12603                                                                  ;GO TO ERROR
12604  062134  170203                  104$:   STFPS   R3                      ;SAVE FPS
12605  062136  012701  003102                  MOV     #RECDST,R1              ;SAVE ACO
12606  062142  174011                          STD     AC0,(R1)                ;
12607  062144  022703  103200                  CMP     #103200,R3              ;VERFIY STATUS
12608  062150  001401                          BEQ     5$                      ;BRANCH IF GOOD
12609  062152  104003                          ERROR   .3              ;FPP ERROR
12610                                                                  ;BAD FPS STATUS
12611  062154  012605                  5$:     MOV     (SP)+,R5                ;GET ERROR PC
12612  062156  020527  062130                  CMP     R5,#103$                ;VERIFY ERROR ADDRESS ON STACK
12613  062162  001401                          BEQ     105$                    ;BRANCH IF GOOD
12614  062164  104003                          ERROR   .3              ;FPP ERROR
12615                                                                  ;BAD ERROR RETURN ON STACK
```

J2


```
12616  062166  005726                  105$:   TST     (SP)+                   ;RESTORE STACK
12617  062170  012704  003504                  MOV     #TAB24,R4               ;POINT TO EXPECTED DATA
12618  062174  004767  050472                  JSR     R7,DATVER
12619  062200  005767  120654                  TST     COUNT                   ;
12620  062204  001401                          BEQ     6$                      ;BRANCH IF  GOOD
12621  062206  104003                          ERROR   .3                  ;FPP ERROR
12622                                                                       ;BAD ACO
12623                                          ;       !-FSRC! <           !ACO!
12624  062210  012704  003524          6$:     MOV     #TAB26,R4               ;POINT TO FSRC DATA
12625  062214  012701  003514                  MOV     #TAB25,R1               ;POINT TO ACO DATA
12626  062220  012702  003200                  MOV     #3200,R2                ;LOAD FPS VALUE
12627  062224  170102                          LDFPS   R2                      ;
12628  062226  012737  000246  000244          MOV     #246,@#FPVEC            ;SETUP FP VECTOR
12629  062234  172411                          LDD     (R1),AC0                ;LOAD AC0 DATA
12630  062236  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12631  062240  170203                          STFPS   R3                      ;SAVE STATUS
12632  062242  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12633  062246  174011                          STD     AC0,(R1)                        ;SAVE AC0
12634  062250  020327  003200                  CMP     R3,#3200                        ;VERIFY STATUS
12635  062254  001401                          BEQ     7$                      ;BRANCH IF GOOD
12636  062256  104003                          ERROR   .3                  ;FPP ERROR
12637                                                                       ;BAD FPS
12638  062260  012704  003534          7$:     MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
12639  062264  004767  050402                  JSR     R7,DATVER                       ;VERIFY DATA
12640  062270  005767  120564                  TST     COUNT                   ;
12641  062274  001401                          BEQ     8$                      ;BRANCH IF GOOD
12642  062276  104003                          ERROR   .3                  ;FPP ERROR
12643                                                                       ;
12644                                          ;       !FSRC!  >           !-AC!
12645  062300  012704  003514          8$:     MOV     #TAB25,R4               ;POINT TO FSRC DATA
12646  062304  012701  003524                  MOV     #TAB26,R1               ;POINT TO AC0 DATA
12647  062310  172411                          LDD     (R1),AC0                ;LOAD AC0 DATA
12648  062312  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12649  062314  170203                          STFPS   R3                      ;SAVE STATUS
12650  062316  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12651  062322  174011                          STD     AC0,(R1)                        ;SAVE AC0
12652  062324  020327  003200                  CMP     R3,@3200                        ;VERIFY STATUS
12653  062330  001401                          BEQ     9$                      ;BRANCH IF GOOD
12654  062332  104003                          ERROR   .3                  ;FPP ERROR
12655                                                                       ;BAD FPS
12656  062334  012704  003534          9$:     MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
12657  062340  004767  050326                  JSR     R7,DATVER                       ;VERIFY DATA
12658  062344  005767  120510                  TST     COUNT                   ;
12659  062350  001401                          BEQ     10$                     ;BRANCH IF GOOD
12660  062352  104003                          ERROR   .3                  ;FPP ERROR
12661                                                                       ;BAD ACO
12662                                          ;       !-FSRC! <           !ACO!
12663  062354  012704  003554          10$:    MOV     #TAB29,R4               ;POINT TO FSRC DATA
12664  062360  012701  003544                  MOV     #TAB28,R1               ;POINT TO AC0 DATA
12665  062364  172411                          LDD     (R1),AC0                ;LOAD AC0 DATA
12666  062366  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12667  062370  170203                          STFPS   R3                      ;SAVE STATUS
12668  062372  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12669  062376  174011                          STD     AC0,(R1)                        ;SAVE AC0
12670  062400  020327  003200                  CMP     R3,#3200                        ;VERIFY STATUS
12671  062404  001401                          BEQ     11$                     ;BRANCH IF GOOD
```

```
 12672  062406  104003                          ERROR   +3              ;FPP ERROR
 12673                                                                   ;BAD FPS
 12674  062410  012704  003564          11$:    MOV     #TAB29A,R4              ;POINT TO EXPECTED DATA
 12675  062414  004767  050252                  JSR     R7,DATVER                       ;VERIFY DATA
 12676  062420  005767  120434                  TST     COUNT                   ;
 12677  062424  001401                          BEQ     12$                     ;BRANCH IF GOOD
 12678  062426  104003                          ERROR   +3              ;FPP ERROR
 12679                                                                   ;
 12680  062430                          12$:
 12681
 12682                                  ;
 12683                                  ;
 12684                                  ;
 12685                                  ;
 12686                                  ;---------------------------------
 12687                                  ;TEST SUB WITH EXP[AC0]=EXP[FSRC]
 12688                                  ;
 12689                                          ;
 12690                                          ;
 12691                                          ;
 12692  062430                          MSB:
 12693
 12694                                  ;
 12695  062430  012702  003200                  MOV     #3200,R2                ;LOAD FPS DATA
 12696  062434  170102                          LDFPS   R2                      ;LOAD FPS
 12697  062436  012704  003454                  MOV     #TAB21,R4               ;POINT TO FSRC DATA
 12698  062442  012701  003102                  MOV     #RECDST,R1              ;POINT TO AC0 RECEIVED DATA TABLE
 12699  062446  172414                          LDD     (R4),AC0                ;LOAD AC0
 12700  062450  173014                          SUBD    (R4),AC0                        ;*TEST INSTRUCTION
 12701  062452  170203                          STFPS   R3                      ;SAVE STATUS
 12702  062454  174011                          STD     AC0,(R1)                ;SAVE AC0 INTO RECDST
 12703  062456  022703  003204                  CMP     #3204,R3                ;VERIFY STATUS
 12704  062462  001401                          BEQ     1$                      ;BRANCH IF GOOD
 12705  062464  104003                          ERROR   +3              ;FPP ERROR
 12706                                                                   ;BAD FPS STATUS
 12707  062466  012704  003254          1$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
 12708  062472  004767  050174                  JSR     R7,DATVER               ;VERIFY AC0
 12709  062476  005767  120356                  TST     COUNT                   ;
 12710  062502  001401                          BEQ     2$                      ;BRANCH IF GOOD
 12711  062504  104003                          ERROR   +3              ;FPP ERROR
 12712                                                                   ;BAD AC0
 12713  062506  012704  003404          2$:     MOV     #TAB14,R4               ;POINT TO FSRC AND AC0 DATA
 12714  062512  172414                          LDD     (R4),AC0                        ;LOAD AC0 DATA
 12715  062514  173014                          SUBD    (R4),AC0                        ;*TEST  INSTRUCTION
 12716  062516  170203                          STFPS   R3                      ;SAVE FPS
 12717  062520  174011                          STD     AC0,(R1)                                ;SAVE AC0 INTO RECDST
 12718  062522  022703  003204                  CMP     #3204,R3                        ;VERIFY FPS
 12719  062526  001401                          BEQ     3$                      ;BRANCH IF GOOD
 12720  062530  104003                          ERROR   +3              ;FPP ERROR
 12721                                                                   ;BAD AC0
 12722  062532  012704  003254          3$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
 12723  062536  004767  050130                  JSR     R7,DATVER                       ;VERIFY AC0
 12724  062542  005767  120312                  TST     COUNT
 12725  062546  001401                          BEQ     4$                      ;BRANCH IF GOOD
 12726  062550  104003                          ERROR   +3              ;FPP ERROR
 12727                                                                   ;BAD AC0
```

```
12728  062552                          4$:
12729
12730                                  ;
12731                                  ;
12732                                  ;
12733                                  ;---------------------------------
12734                                  ;TEST NORMALIZE
12735                                  ;
12736                                          ;
12737                                          ;
12738                                          ;
12739  062552                          MNRM:
12740
12741                                  ;
12742  062552  012702  003200                  MOV     #3200,R2                ;LOAD FPS
12743  062556  170102                          LDFPS   R2                      ;
12744  062560  012705  003604                  MOV     #TAB31,R5               ;POINT TO FSRC DATA
12745  062564  012701  003574                  MOV     #TAB30,R1               ;POINT TO ACO DATA
12746  062570  172411                          LDD     (R1),AC0                ;LOAD AC0
12747  062572  173015                          SUBD    (R5),AC0                ;*TEST INSTRUCTION
12748                                                                          ;1 LEFT SHIFT
12749  062574  170203                          STFPS   R3                      ;SAVE STATUS
12750  062576  012704  003102                  MOV     #RECDST,R4              ;POINT TO RECDATA
12751  062602  174014                          STD     AC0,(R4)                        ;SAVE AC0
12752  062604  012701  003634                  MOV     #TAB34,R1               ;POINT TO EXPECTED DATA
12753  062610  004767  050056                  JSR     R7,DATVER                       ;VERIFY DATA
12754  062614  005767  120240                  TST     COUNT
12755  062620  001401                          BEQ     1$                      ;BRANCH IF GOOD
12756  062622  104003                          ERROR   .+3                 ;FPP ERROR
12757                                                                      ;
12758  062624  012701  003614          1$:     MOV     #TAB32,R1                       ;AC0 DATA
12759  062630  012705  003624                  MOV     #TAB33,R5                               ;FSRC DATA
12760  062634  172411                          LDD     (R1),AC0                                ;LOAD AC0
12761  062636  173015                          SUBD    (R5),AC0                ;*TST INSTRUCTION
12762                                                                          ;56 LEFT SHIFTS
12763  062640  012701  003102                  MOV     #RECDST,R1                              ;SAVE DATA
12764  062644  174011                          STD     AC0,(R1)                        ;
12765  062646  004767  050020                  JSR     R7,DATVER
12766  062652  005767  120202                  TST     COUNT
12767  062656  001401                          BEQ     2$                      ;
12768  062660  104003                          ERROR   .+3                 ;FPP ERROR
12769                                                                      ;
12770  062662                          2$:
12771
12772                                  ;
12773                                  ;
12774                                  ;---------------------------------
12775                                  ;TEST ADDD WITH OVERFLOW AND UNDERFLOW
12776                                  ;
12777                                          ;
12778                                          ;
12779                                          ;
12780  062662                          MUVAD:
12781
12782                                  ;
12783  062662  012702  000200                  MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
```

```
12784  062666  170102                          LDFPS   R2                      ;LOAD FPS
12785  062670  012704  003644                  MOV     #TAB40,R4                       ;POINT TO FSRC DATA
12786  062674  012701  003644                  MOV     #TAB40,R1                       ;POINT TO ACO DATA
12787  062700  172411                          LDD     (R1),AC0                        ;LOAD ACO WITH TEST DATA
12788  062702  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12789  062704  170203                          STFPS   R3                      ;SAVE FPS
12790  062706  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12791  062712  174011                          STD     AC0,(R1)                ;SAVE ACO  RESULT
12792  062714  022703  000206                  CMP     #206,R3                 ;VERIFY STATUS
12793  062720  001401                          BEQ     1$                      ;BRANCH IF GOOD
12794  062722  104003                          ERROR   .3              ;FPP ERROR
12795                                                                  ;BAD FPS
12796  062724  012704  003254          1$:     MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
12797  062730  004767  047736                  JSR     R7,DATVER               ;VERIFY DATA
12798  062734  005767  120120                  TST     COUNT
12799  062740  001401                          BEQ     2$                      ;BRANCH IF GOOD
12800  062742  104003                          ERROR   .3              ;FPP ERROR
12801                                                                  ;BAD ACO
12802                                  ;OVERFLOW TRAPS ENABLED
12803  062744  012702  001200          2$:     MOV     #1200,R2                        ;SETUP FLOATING POINT STATUS
12804  062750  170102                          LDFPS   R2                      ;LOAD FPS
12805  062752  012704  003644                  MOV     #TAB40,R4                       ;POINT TO FSRC DATA
12806  062756  012701  003644                  MOV     #TAB40,R1                       ;POINT TO ACO DATA
12807  062762  172411                          LDD     (R1),AC0                        ;LOAD ACO WITH TEST DATA
12808  062764  012737  063000  000244          MOV     #3$,@#FPVEC             ;CHANGE TRAP VECTOR
12809  062772  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12810  062774  170000                  23$:    CFCC
12811  062776  104003                          ERROR   .3              ;FPP ERROR
12812                                                                  ;FAILED TO TRAP ON OVERFLOW
12813  063000  170203                  3$:     STFPS   R3                      ;SAVE FPS
12814  063002  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12815  063006  174011                          STD     AC0,(R1)                ;SAVE ACO  RESULT
12816  063010  022703  101206                  CMP     #101206,R3                      ;VERIFY STATUS
12817  063014  001401                          BEQ     4$                      ;BRANCH IF GOOD
12818  063016  104003                          ERROR   .3              ;FPP ERROR
12819                                                                  ;BAD FPS
12820  063020  012600                  4$:     MOV     (SP)+,R0                ;CHECK STORED PC
12821  063022  022700  062774                  CMP     #23$,R0                 ;
12822  063026  001401                          BEQ     5$                      ;BRANCH IF RETURN ADDRESS IS GOOD
12823  063030  104003                          ERROR   .3              ;FPP ERROR
12824                                                                  ;BAD RETURN ADDRESS
12825  063032  012600                  5$:     MOV     (SP)+,R0                ;CLEAN UP STACK
12826  063034  012704  003254                  MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
12827  063040  004767  047626                  JSR     R7,DATVER               ;VERIFY DATA
12828  063044  005767  120010                  TST     COUNT
12829  063050  001401                          BEQ     7$                      ;BRANCH IF GOOD
12830  063052  104003                          ERROR   .3              ;FPP ERROR
12831                                                                  ;BAD ACO
12832                                  ;UNDERFLOW TRAPS DISABLED
12833  063054  012702  000200          7$:     MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
12834  063060  170102                          LDFPS   R2                      ;LOAD FPS
12835  063062  012737  132636  000244          MOV     #WLDTRP,@#FPVEC                 ;REPLACE WILD TRAP VECTOR
12836  063070  012704  003274                  MOV     #TAB7,R4                        ;POINT TO FSRC DATA
12837  063074  012701  003654                  MOV     #TAB41,R1                       ;POINT TO ACO DATA
12838  063100  172411                          LDD     (R1),AC0                        ;LOAD ACO WITH TEST DATA
12839  063102  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
```

```
12840  063104  170203                          STFPS   R3              ;SAVE FPS
12841  063106  012701  003102                  MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
12842  063112  174011                          STD     AC0,(R1)        ;SAVE AC0  RESULT
12843  063114  022703  000204                  CMP     #204,R3         ;VERIFY STATUS
12844  063120  001401                          BEQ     8$              ;BRANCH IF GOOD
12845  063122  104003                          ERROR   .3              ;FPP ERROR
12846                                                                  ;BAD FPS
12847  063124  012704  003254          8$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
12848  063130  004767  047536                  JSR     R7,DATVER               ;VERIFY DATA
12849  063134  005767  117720                  TST     COUNT
12850  063140  001401                          BEQ     9$              ;BRANCH IF GOOD
12851  063142  104003                          ERROR   .3              ;FPP ERROR
12852                                                                  ;BAD AC0
12853                          ;UNDERFLOW TRAPS ENABLED
12854  063144  012702  002200          9$:     MOV     #2200,R2                ;SETUP FLOATING POINT STATUS
12855  063150  170102                          LDFPS   R2              ;LOAD FPS
12856  063152  012737  063200  000244          MOV     #11$,@#FPVEC    ;REPOSITION TRAP VECTOR
12857  063160  012704  003274                  MOV     #TAB7,R4                ;POINT TO FSRC DATA
12858  063164  012701  003654                  MOV     #TAB41,R1               ;POINT TO AC0 DATA
12859  063170  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
12860  063172  172014                          ADDD    (R4),AC0        ;*TEST INSTRUCTION
12861  063174  170000                  10$:    CFCC                    ;COPY FPP CC
12862  063176  104003                          ERROR   .3              ;FPP ERROR
12863                                                                  ;FAILED TO TRAP ON UNDERFLOW
12864  063200  170203                  11$:    STFPS   R3              ;SAVE FPS
12865  063202  012701  003102                  MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
12866  063206  174011                          STD     AC0,(R1)        ;SAVE AC0  RESULT
12867  063210  022703  102210                  CMP     #102210,R3              ;VERIFY STATUS
12868  063214  001401                          BEQ     12$             ;BRANCH IF GOOD
12869  063216  104003                          ERROR   .3              ;FPP ERROR
12870                                                                  ;BAD FPS
12871  063220  012605                  12$:    MOV     (SP)+,R5                ;GET ERROR PC
12872  063222  020527  063174                  CMP     R5,#10$                 ;VERIFY ERROR ADDRESS ON STACK
12873  063226  001401                          BEQ     13$             ;BRANCH IF GOOD
12874  063230  104003                          ERROR   .3              ;FPP ERROR
12875                                                                  ;BAD ERROR RETURN ON STACK
12876  063232  005726                  13$:    TST     (SP)+                   ;RESTORE STACK
12877  063234  012704  003244                  MOV     #TAB5A,R4                       ;POINT TO EXPECTED DATA
12878  063240  004767  047426                  JSR     R7,DATVER                       ;VERIFY DATA
12879  063244  005767  117610                  TST     COUNT
12880  063250  001401                          BEQ     14$             ;BRANCH IF GOOD
12881  063252  104003                          ERROR   .3              ;FPP ERROR
12882                                                                  ;BAD AC0
12883                          ;UNDERFLOW WITH TRAPS DISBLED - NON-ZERO RESULT
12884  063254  012702  000200          14$:    MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
12885  063260  170102                          LDFPS   R2              ;LOAD FPS
12886  063262  012737  132636  000244          MOV     #%LDTRP,@#FPVEC         ;RESTORE TRAP VECTOR
12887  063270  012704  003654                  MOV     #TAB41,R4               ;POINT TO FSRC DATA
12888  063274  012701  003664                  MOV     #TAB42,R1               ;POINT TO AC0 DATA
12889  063300  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
12890  063302  172014                          ADDD    (R4),AC0        ;*TEST INSTRUCTION
12891  063304  170203                          STFPS   R3              ;SAVE FPS
12892  063306  012701  003102                  MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
12893  063312  174011                          STD     AC0,(R1)        ;SAVE AC0  RESULT
12894  063314  022703  000204                  CMP     #204,R3         ;VERIFY STATUS
12895  063320  001401                          BEQ     15$             ;BRANCH IF GOOD
```

```
12896  063322  104003                          ERROR   ·3              ;FPP ERROR
12897                                                                   ;BAD FPS
12898  063324  012704  003254          15$:    MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
12899  063330  004767  047336                  JSR     R7,DATVER               ;VERIFY DATA
12900  063334  005767  117520                  TST     COUNT
12901  063340  001401                          BEQ     16$             ;BRANCH IF GOOD
12902  063342  104003                          ERROR   ·3              ;FPP ERROR
12903                                                                   ;BAD ACO
12904                                  ;UNERFLOW WITH TRAPS ENABLED - NON-ZERO  RESULT
12905  063344  012702  102200          16$:    MOV     #102200,R2              ;SETUP FLOATING POINT STATUS
12906  063350  170102                          LDFPS   R2              ;LOAD FPS
12907  063352  012737  063400  000244          MOV     #18$,@#FPVEC            ;RESTORE TRAP VECTOR
12908  063360  012704  003654                  MOV     #TAB41,R4               ;POINT TO FSRC DATA
12909  063364  012701  003664                  MOV     #TAB42,R1               ;POINT TO ACO DATA
12910  063370  172411                          LDD     (R1),AC0                ;LOAD ACO WITH TEST DATA
12911  063372  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12912  063374  170000          17$:    CFCC
12913  063376  104003                          ERROR   ·3              ;FPP ERROR
12914                                                                   ;NO TRAP ON UNDERFLOW
12915  063400  170203          18$:    STFPS   R3                      ;SAVE FPS
12916  063402  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12917  063406  174011                          STD     AC0,(R1)                ;SAVE ACO  RESULT
12918  063410  012600                          MOV     (SP)+,R0                ;SAVE STACK CONTENTS
12919  063412  005726                          TST     (SP)+                   ;CLEAN UP STACK
12920  063414  022700  063374                  CMP     #17$,R0                 ;VERIFY RETURN ADDRESS
12921  063420  001401                          BEQ     19$                     ;BRANCH IF GOOD
12922  063422  104003                          ERROR   ·3              ;FPP ERROR
12923                                                                   ;BAD RETURN ADDRESS
12924  063424  022703  102204          19$:    CMP     #102204,R3              ;VERIFY STATUS
12925  063430  001401                          BEQ     20$                     ;BRANCH IF GOOD
12926  063432  104003                          ERROR   ·3              ;FPP ERROR
12927                                                                   ;BAD FPS
12928  063434  012704  003674          20$:    MOV     #TAB43,R4               ;POINT TO EXPECTED DATA
12929  063440  004767  047226                  JSR     R7,DATVER               ;VERIFY DATA
12930  063444  005767  117410                  TST     COUNT
12931  063450  001401                          BEQ     21$             ;BRANCH IF GOOD
12932  063452  104003                          ERROR   ·3              ;FPP ERROR
12933                                                                   ;BAD ACO
12934  063454                          21$:
12935
12936
12937                                  ;
12938                                  ;
12939                                  ;------------------------- -------
12940                                  ;TEST LDCFD, LDCDF
12941                                  ;
12942                                          ;
12943                                          ;
12944                                          ;
12945  063454                          MLDC:
12946
12947                                  ;
12948                                  ;TRUNCATE
12949  063454  012702  000300                  MOV     #300,R2                 ;SETUP FLOATING POINT STATUS
12950  063460  170102                          LDFPS   R2                      ;LOAD FPS
12951  063462  012704  003704                  MOV     #TAB45,R4                       ;POINT TO FSRC DATA
```

```
 12952  063466  012701  003254                MOV     #TAB6,R1                ;POINT TO ACO DATA
 12953  063472  172411                        LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
 12954  063474  177424                        LDCFD   (R4)+,AC0               ;*TEST INSTRUCTION
 12955  063476  012701  003102                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
 12956  063502  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
 12957  063504  022704  003710                CMP     #TAB45+4,R4             ;VERIFY AUTO-INC
 12958  063510  001401                        BEQ     1$                      ;BRANCH IF GOOD AUTO-INC
 12959  063512  104003                        ERROR   +3                      ;FPP ERROR
 12960                                                                        ;BAD AUTO-INC
 12961  063514  012704  003714        1$:     MOV     #TAB46,R4               ;POINT TO EXPECTED DATA
 12962  063520  004767  047146                JSR     R7,DATVER               ;VERIFY DATA
 12963  063524  005767  117330                TST     COUNT
 12964  063530  001401                        BEQ     2$                      ;BRANCH IF GOOD
 12965  063532  104003                        ERROR   +3                      ;FPP ERROR
 12966                                                                        ;BAD AC0
 12967                                ;AUTO-INC DOUBLE MODE
 12968  063534  005002                2$:     CLR     R2                      ;SETUP FLOATING POINT STATUS
 12969  063536  170102                        LDFPS   R2                      ;LOAD FPS
 12970  063540  012704  003704                MOV     #TAB45,R4               ;POINT TO FSRC DATA
 12971  063544  012701  003424                MOV     #TAB16,R1               ;POINT TO ACO DATA
 12972  063550  172411                        LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
 12973  063552  177424                        LDCDF   (R4)+,AC0               ;*TEST INSTRUCTION
 12974  063554  020427  003714                CMP     R4,#TAB45+10            ;VERIFY AUTO-INC
 12975  063560  001401                        BEQ     3$                      ;BRANCH IF GOOD
 12976  063562  104003                        ERROR   +3                      ;FPP ERROR
 12977                                                                        ;BAD AUTO-INC ON DOUBLE
 12978  063564  170203                3$:     STFPS   R3                      ;SAVE FPS
 12979  063566  012701  003102                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
 12980  063572  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
 12981  063574  022703  000000                CMP     #0,R3                   ;VERIFY STATUS
 12982  063600  001401                        BEQ     4$                      ;BRANCH IF GOOD
 12983  063602  104003                        ERROR   +3                      ;FPP ERROR
 12984                                                                        ;BAD FPS
 12985  063604  012704  003754        4$:     MOV     #TAB49,R4               ;POINT TO EXPECTED DATA
 12986  063610  004767  047056                JSR     R7,DATVER               ;VERIFY DATA
 12987  063614  005767  117240                TST     COUNT
 12988  063620  001401                        BEQ     5$                      ;BRANCH IF GOOD
 12989  063622  104003                        ERROR   +3                      ;FPP ERROR
 12990                                                                        ;BAD AC0
 12991                                ;LDCFD GR7
 12992  063624  012702  000200        5$:     MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
 12993  063630  170102                        LDFPS   R2                      ;LOAD FPS
 12994  063632  005003                        CLR     R3
 12995  063634  177427  043243                LDCFD   #5203,AC0               ;*TEST INSTRUCTION
 12996  063640  005203                        INC     R3
 12997  063642  005203                        INC     R3
 12998  063644  005203                        INC     R3                      ;IF LDCFD WORKED, R3 SHOULD=3
 12999  063646  022703  000003                CMP     #3,R3                   ;VERIFY CORRECT PROGRAM FLOW
 13000  063652  001401                        BEQ     6$                      ;BRANCH IF GOOD
 13001  063654  104003                        ERROR   +3                      ;FPP ERROR
 13002                                                                        ;BAD PROGRAM FLOW
 13003                                ;NEGATIVE OPERANDS
 13004  063656  012702  000200        6$:     MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
 13005  063662  170102                        LDFPS   R2                      ;LOAD FPS
 13006  063664  012704  003724                MOV     #TAB47,R4               ;POINT TO FSRC DATA
 13007  063670  012701  003704                MOV     #TAB45,R1               ;POINT TO ACO DATA
```

```
13008  063674  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13009  063676  177414                          LDCFD   (R4),AC0                ;*TEST INSTRUCTION
13010  063700  170203                          STFPS   R3                      ;SAVE FPS
13011  063702  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13012  063706  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13013  063710  022703  000210                  CMP     #210,R3                 ;VERIFY STATUS
13014  063714  001401                          BEQ     7$                      ;BRANCH IF GOOD
13015  063716  104003                          ERROR   .3                      ;FPP ERROR
13016                                                                          ;BAD FPS
13017  063720  012704  003744          7$:     MOV     #TAB48,R4               ;POINT TO EXPECTED DATA
13018  063724  004767  046742                  JSR     R7,DATVER               ;VERIFY DATA
13019  063730  005767  117124                  TST     COUNT
13020  063734  001401                          BEQ     8$                      ;BRANCH IF GOOD
13021  063736  104003                          ERROR   .3                      ;FPP ERROR
13022                                                                          ;BAD AC0
13023                                  ;LOAD A ZERO
13024  063740  012702  000200          8$:     MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13025  063744  170102                          LDFPS   R2                      ;LOAD FPS
13026  063746  012704  003254                  MOV     #TAB6,R4                ;POINT TO FSRC DATA
13027  063752  012701  003744                  MOV     #TAB48,R1               ;POINT TO AC0 DATA
13028  063756  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13029  063760  177414                          LDCFD   (R4),AC0                ;*TEST INSTRUCTION
13030  063762  170203                          STFPS   R3                      ;SAVE FPS
13031  063764  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13032  063770  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13033  063772  022703  000204                  CMP     #204,R3                 ;VERIFY STATUS
13034  063776  001401                          BEQ     9$                      ;BRANCH IF GOOD
13035  064000  104003                          ERROR   .3                      ;FPP ERROR
13036                                                                          ;BAD FPS
13037  064002  012704  003254          9$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
13038  064006  004767  046660                  JSR     R7,DATVER               ;VERIFY DATA
13039  064012  005767  117042                  TST     COUNT
13040  064016  001401                          BEQ     10$                     ;BRANCH IF GOOD
13041  064020  104003                          ERROR   .3                      ;FPP ERROR
13042                                                                          ;BAD AC0
13043  064022                          10$:
13044
13045                                  ;
13046                                  ;
13047                                  ;
13048                                  ;---------------------- ----------
13049                                  ;TEST CMPD
13050                                  ;
13051                                          ;
13052                                          ;
13053                                          ;
13054  064022                          MCMPD:
13055
13056                                  ;
13057                                  ;  CMPD WITH FRSC=AC0=0
13058  064022  005037  002776                  CLR     @#FLAG                  ;SIGNAL THAT AC0 REMAINS CONSTANT
13059  064026  004767  000152                  JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13060  064032  000000  000000  000000          .WORD   0,0,0,0                 ;AC0 AT START
13061  064040  000000
13062  064042  000000  000000  000000          .WORD   0,0,0,0                 ;FSRC AT START
13063  064050  000000
```

E 7


```
13064  064052  000200                          .WORD   200                     ;FPS AT START (D)
13065  064054  000204                          .WORD   204                     ;FPS AT END
13066                                  ;  CMPD WITH EXP[FSRC]=0, EXP[AC0]=0
13067  064056  012737  000001  002776          MOV     #1,@#FLAG               ;SIGNAL THAT AC0 WILL = 0
13068  064064  004767  000114                  JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13069  064070  000000  000000  000000          .WORD   0,0,0,125252            ;AC0 AT START
13070  064076  125252
13071  064100  000100  000022  000123          .WORD   100,22,123,123          ;FSRC AT START
13072  064106  000123
13073  064110  000200                          .WORD   200                     ;FPS AT START (D)
13074  064112  000204                          .WORD   204                     ;FPS AT END
13075                                  ;  CMPD FSRC>EXP[AC0]=0
13076  064114  005037  002776                  CLR     @#FLAG                  ;AC0 REMAINS UNCHANGED
13077  064120  004767  000060                  JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13078  064124  000400  012346  012346          .WORD   400,12346,12346,23      ;AC0 AT START
13079  064132  000023
13080  064134  000200  000000  000000          .WORD   200,0,0,0               ;FSRC AT START
13081  064142  000000
13082  064144  000200                          .WORD   200                     ;FPS AT START (D)
13083  064146  000210                          .WORD   210                     ;FPS AT END
13084                                  ; CMPD FSRC=AC0>0
13085  064150  004767  000030                  JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13086  064154  077777  177777  177777          .WORD   77777,-1,-1,-1          ;AC0 AT START
13087  064162  177777
13088  064164  077777  177777  177777          .WORD   77777,-1,-1,-1          ;FSRC AT START
13089  064172  177777
13090  064174  000200                          .WORD   200                     ;FPS AT START (D)
13091  064176  000204                          .WORD   204                     ;FPS AT END
13092  064200  000167  000116                  JMP     HOP44                   ;HOP OVER SUBROUTINE
13093
13094                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13095                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13096                                  ;COMPARE ROUTINE DATA TABLES
13097                                  ;
13098                                  ;               AC0
13099                                  ;               FSRC
13100                                  ;               FPS BEFORE EXECUTION
13101                                  ;               FPS AFTER EXECUTION
13102                                  ;               (FEC)
13103                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13104                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13105                                  ;
13106                                  ;
13107  064204  012605                  CMPRTN: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13108  064206  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13109  064212  170102                          LDFPS   R2                      ;LOAD FPS
13110  064214  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13111  064216  062704  000010                  ADD     #10,R4                  ;
13112  064222  010501                          MOV     R5,R1           ;POINT TO AC0 DATA
13113  064224  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13114  064226  016502  000020                  MOV     20(R5),R2               ;GET TEST FPS
13115  064232  170102                          LDFPS   R2                      ;LOAD TEST FPS
13116  064234  173414                  10:     CMPD    (R4),AC0                ;*TEST INSTRUCTION
13117  064236  170203                          STFPS   R3                      ;SAVE FPS
13118  064240  012702  000200                  MOV     #200,R2         ;SET FPP TO DOUBLE
13119  064244  170102                          LDFPS   R2
```

F3


```
13120  064246  012701  003102              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13121  064252  174011                      STD     AC0,(R1)                ;SAVE AC0  RESULT
13122  064254  026503  000022              CMP     22(R5),R3               ;VERIFY STATUS
13123  064260  001401                      BEQ     2$                      ;BRANCH IF GOOD
13124  064262  104003                      ERROR   .3              ;FPP ERROR
13125                                                                ;BAD FPS
13126  064264  005737  002776      2$:     TST     @#FLAG                  ;SEE IF AC0 REMAINS UNCHANGED
13127  064270  001403                      BEQ     3$                      ;BRANCH IF AC0 STAYS THE SAME
13128  064272  012704  003254              MOV     #TAB6,R4                ;AC0=0
13129  064276  000401                      BR      4$                      ;GO VERIFY  DATA
13130  064300  010504              3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13131  064302  004767  046364      4$:     JSR     R7,DATVER                       ;VERIFY DATA
13132  064306  005767  116546              TST     COUNT
13133  064312  001401                      BEQ     5$                      ;BRANCH IF GOOD
13134  064314  104003                      ERROR   .3              ;FPP ERROR
13135                                                                ;BAD AC0
13136  064316  000165  000024      5$:     JMP     24(R5)                  ;RETURN
13137  064322                      HOP44:
13138
13139
13140
13141                               ;
13142                               ;
13143                               ;---- -------- ---------------- -
13144                               ;TEST DIVF
13145                               ;
13146                                       ;
13147                                       ;
13148                                       ;
13149  064322                      MDIVF:
13150
13151                               ;
13152                               ;1/EXP[AC]=FSRC=0
13153  064322  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPT, BUT FEC
13154  064330  004767  000706                  JSR     R7,DVFSUB               ;DO TEST
13155  064334  000100  000027                  .WORD   100,27          ;AC0
13156  064340  000000  000000                  .WORD   0,0             ;FSRC
13157  064344  000100  000027                  .WORD   100,27          ;RESULT
13158  064350  040000                          .WORD   40000           ; TEST FPS
13159  064352  140000                          .WORD   140000          ;RESULT FPS
13160  064354  000004                          .WORD   4               ;FEC
13161                               ;2/AC=EXP[FSRC]=0
13162                               ;TRAPS ENABLED
13163  064356  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
13164  064364  004767  000652                  JSR     R7,DVFSUB               ;DO TEST
13165  064370  000000  000000                  .WORD   0,0             ;AC0
13166  064374  000100  000000                  .WORD   100,0           ;FSRC
13167  064400  000000  000000                  .WORD   0,0             ;RESULT
13168  064404  000000                          .WORD   0               ; TEST FPS
13169  064406  100000                          .WORD   100000          ;RESULT FPS
13170  064410  000004                          .WORD   4               ;FEC
13171                               ;3/FSRC>AC0=0
13172  064412  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13173  064416  004767  000620                  JSR     R7,DVFSUB               ;DO TEST
13174  064422  000177  000234                  .WORD   177,234         ;AC0
13175  064426  004100  000000                  .WORD   4100,0          ;FSRC
```

```
 13176  064432  000000  000000                 .WORD   0,0             ;RESULT
 13177  064436  007400                         .WORD   7400            ; TEST FPS
 13178  064440  007404                         .WORD   7404            ;RESULT FPS
 13179                                 ;4/ACO>EXP[FSRC]=0
 13180  064442  012737  000001  002776         MOV     #1,@#FLAG               ;INTERRUPT
 13181  064450  004767  000566                 JSR     R7,DVFSUB               ;DO TEST
 13182  064454  040200  104210                 .WORD   40200,104210    ;ACO
 13183  064460  000125  025252                 .WORD   125,25252       ;FSRC
 13184  064464  040200  104210                 .WORD   40200,104210    ;RESULT
 13185  064470  007557                         .WORD   7557            ; TEST FPS
 13186  064472  107557                         .WORD   107557          ;RESULT FPS
 13187  064474  000004                         .WORD   4               ;FEC
 13188                                 ;5/EXP[AC]=EXP[FSRC]
 13189  064476  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13190  064502  004767  000534                 JSR     R7,DVFSUB               ;DO TEST
 13191  064506  077760  177777                 .WORD   77760,-1                ;ACO
 13192  064512  077760  000000                 .WORD   77760,0         ;FSRC
 13193  064516  040200  104210                 .WORD   40200,104210    ;RESULT
 13194  064522  007414                         .WORD   7414            ; TEST FPS
 13195  064524  007400                         .WORD   7400            ;RESULT FPS
 13196                                 ;6/AC=FSRC
 13197  064526  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13198  064532  004767  000504                 JSR     R7,DVFSUB               ;DO TEST
 13199  064536  052525  052525                 .WORD   52525,52525             ;ACO
 13200  064542  052525  052525                 .WORD   52525,52525             ;FSRC
 13201  064546  040200  000000                 .WORD   40200,0         ;RESULT
 13202  064552  007400                         .WORD   7400            ; TEST FPS
 13203  064554  007400                         .WORD   7400            ;RESULT FPS
 13204                                 ;7/FSRC>0<ACO, ROUND
 13205  064556  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13206  064562  004767  000454                 JSR     R7,DVFSUB               ;DO TEST
 13207  064566  077777  125252                 .WORD   77777,125252    ;ACO
 13208  064572  040300  000000                 .WORD   40300,0         ;FSRC
 13209  064576  077652  070707                 .WORD   77652,070707            ;RESULT
 13210  064602  007400                         .WORD   7400            ; TEST FPS
 13211  064604  007400                         .WORD   7400            ;RESULT FPS
 13212                                 ;8/AC>0<FSRC
 13213  064606  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13214  064612  004767  000424                 JSR     R7,DVFSUB               ;DO TEST
 13215  064616  055377  177777                 .WORD   55377,-1        ;ACO
 13216  064622  055300  000000                 .WORD   55300,0         ;FSRC
 13217  064626  040252  125252                 .WORD   40252,125252    ;RESULT
 13218  064632  000000                         .WORD   0               ; TEST FPS
 13219  064634  000000                         .WORD   0               ;RESULT FPS
 13220                                 ;9/FSRC>AC>0
 13221  064636  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13222  064642  004767  000374                 JSR     R7,DVFSUB               ;DO TEST
 13223  064646  064600  000001                 .WORD   64600,1         ;ACO
 13224  064652  066600  000000                 .WORD   66600,0         ;FSRC
 13225  064656  036200  000001                 .WORD   36200,1         ;RESULT
 13226  064662  000000                         .WORD   0               ; TEST FPS
 13227  064664  000000                         .WORD   0               ;RESULT FPS
 13228                                 ;10/AC>FSRC>0
 13229  064666  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 13230  064672  004767  000344                 JSR     R7,DVFSUB               ;DO TEST
 13231  064676  012345  156024                 .WORD   12345,156024    ;ACO
```

```
13232  064702  005600  000000                .WORD   05600,0         ;FSRC
13233  064706  044745  156024                .WORD   44745,156024            ;RESULT
13234  064712  000017                        .WORD   17              ; TEST FPS
13235  064714  000000                        .WORD   0               ;RESULT FPS
13236                              ;11/FSRC<0
13237  064716  005037  002776                CLR     @#FLAG          ;NO INTERRUPT
13238  064722  004767  000314                JSR     R7,DVFSUB               ;DO TEST
13239  064726  040422  101010                .WORD   40422,101010    ;ACO
13240  064732  140511  101010                .WORD   140511,101010   ;FSRC
13241  064736  140072  020167                .WORD   140072,20167            ;RESULT
13242  064742  000057                        .WORD   57              ; TEST FPS
13243  064744  000050                        .WORD   50              ;RESULT FPS
13244                              ;12/AC<0
13245  064746  005037  002776                CLR     @#FLAG          ;NO INTERRUPT
13246  064752  004767  000264                JSR     R7,DVFSUB               ;DO TEST
13247  064756  160077  000101                .WORD   160077,101      ;ACO
13248  064762  040417  177777                .WORD   40417,-1        ;FSRC
13249  064766  157651  143527                .WORD   157651,143527           ;RESULT
13250  064772  000007                        .WORD   7               ; TEST FPS
13251  064774  000010                        .WORD   10              ;RESULT FPS
13252                              ;13/TRUNCATE TEST
13253  064776  005037  002776                CLR     @#FLAG          ;NO INTERRUPT
13254  065002  004767  000234                JSR     R7,DVFSUB               ;DO TEST
13255  065006  060100  000177                .WORD   60100,177       ;ACO
13256  065012  040300  000000                .WORD   40300,0         ;FSRC
13257  065016  060000  000124                .WORD   60000,124               ;RESULT
13258  065022  000040                        .WORD   40              ; TEST FPS
13259  065024  000040                        .WORD   40              ;RESULT FPS
13260                              ;14/ROUND TEST
13261
13262  065026  005037  002776                CLR     @#FLAG          ;NO INTERRUPT
13263  065032  004767  000204                JSR     R7,DVFSUB               ;DO TEST
13264  065036  060100  000177                .WORD   60100,177       ;ACO
13265  065042  040300  000000                .WORD   40300,0 ;FSRC
13266  065046  060000  000125                .WORD   60000,125               ;RESULT
13267  065052  000000                        .WORD   0               ; TEST FPS
13268  065054  000000                        .WORD   0               ;RESULT FPS
13269                              ;15/OVERFLOW, INTERRUPTS ENABLED
13270  065056  012737  000001  002776        MOV     #1,@#FLAG               ;INTERRUPT
13271  065064  004767  000152                JSR     R7,DVFSUB               ;DO TEST
13272  065070  177700  000000                .WORD   177700,0                ;ACO
13273  065074  000200  000000                .WORD   200,0           ;FSRC
13274  065100  137700  000000                .WORD   137700,0                ;RESULT
13275  065104  001100                        .WORD   1100            ; TEST FPS
13276  065106  101112                        .WORD   101112          ;RESULT FPS
13277  065110  000010                        .WORD   10              ;FEC
13278                              ;16/OVERFLOW, TRAPS DISABLED
13279  065112  012737  000002  002776        MOV     #2,@#FLAG               ;NO INTERRUPT
13280  065120  004767  000116                JSR     R7,DVFSUB               ;DO TEST
13281  065124  000200  000000                .WORD   200,0           ;ACO
13282  065130  177700  000000                .WORD   177700,0                ;FSRC
13283  065134  000000  000000                .WORD   0,0             ;RESULT
13284  065140  041100                        .WORD   41100           ; TEST FPS
13285  065142  041104                        .WORD   41104           ;RESULT FPS
13286  065144  000010                        .WORD   10              ;FEC OVERFLOW
13287                              ;17/UNDERFLOW, TRAPS ENABLED, UV RESULT
```

```
065146  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
065154  004767  000062                  JSR     R7,DVFSUB               ;DO TEST
065160  100200  000000                  .WORD   100200,0                ;AC0
065164  040377  177777                  .WORD   40377,-1                ;FSRC
065170  100000  000001                  .WORD   100000,1                ;RESULT
065174  002000                          .WORD   2000            ; TEST FPS
065176  102014                          .WORD   102014          ;RESULT FPS
065200  000012                          .WORD   12              ;FEC
                                ;18/UNDERFLOW,TRAPS ENABLED, ROUND
065202  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
065210  004767  000026                  JSR     R7,DVFSUB               ;DO TEST
065214  030325  025252                  .WORD   30325,25252             ;AC0
065220  076777  023456                  .WORD   76777,23456             ;FSRC
065224  071525  157716                  .WORD   71525,157716            ;RESULT
065230  002537                          .WORD   2537            ; TEST FPS
065232  102500                          .WORD   102500          ;RESULT FPS
065234  000012                          .WORD   12              ;FEC
                                ;

065236  000167  000212                  JMP     HOP10           ;GO TO NEXT TEST
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;DIVF SUBROUTINE:
                                ;               AC0
                                ;               FSRC
                                ;               FPS BEFORE EXECUTION
                                ;               FPS AFTER EXECUTION
                                ;               (FEC)
                                ;
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
065242  012605                  DVFSUB: MOV     (SP)+,R5                        ; RETURN ADDRESS TO USE AS POINTER
065244  012737  065320  000244          MOV     #50$,@#FPVEC                    ;REDIRECT TRAP VECTOR
065252  012702  000200                  MOV     #200,R2                         ;SET TO DOUBLE MODE FOR LOAD
065256  170102                          LDFPS   R2                              ;LOAD FPS
065260  010504                          MOV     R5,R4                           ;POINT TO FSRC DATA
065262  062704  000004                  ADD     #4,R4
065266  172415                          LDD     (R5),AC0                        ;LOAD AC0 WITH TEST DATA
065270  016502  000014                  MOV     14(R5),R2                       ;GET TEST FPS
065274  170102                          LDFPS   R2                              ;LOAD TEST FPS
                                ;
065276  174414                          DIVF    (R4),AC0                        ;*TEST INSTRUCTION
065300  170001                  1$:     SETF                                    ;WAIT FOR POSSIBLE FPA TRAP.
                                ;
                                ;INSTRUCTION DIDNT TRAP
065302  032737  000001  002776          BIT     #1,@#FLAG                               ;VERIFY A NO TRAP CONDITION
065310  001420                          BEQ     2$                              ;BRANCH IF GOOD
065312  104003                          ERROR   +3                      ;FPP ERROR
                                                                        ;INSTRUCTION SHOULD HAVE TRAPPED
065314  000167  000032                  JMP     2$                              ;REJOIN CODE
                                ;
                                ;INSTRUCTION TRAPPED
065320  032737  000001  002776  50$:    BIT     #1,@#FLAG                       ;SEE IF EXPECTING A TRAP
065326  001003                          BNE     51$                             ;BRANCH IF EXPECTING A TRAP
065330  104003                          ERROR   +3                      ;FPP ERROR
```

```
13344
13345  065332  000167  000014                          JMP     2$                      ;INSTRUCTION WASNT SUPPOSE TO TRAP
13346  065336  012604                          51$:    MOV     (SP)+,R4                ;REJOIN CODE
13347  065340  005726                                  TST     (SP)+                   ;SEE IF PC = INSTRUCTION
13348  065342  022704  065300                          CMP     #1$,R4                  ;CLEAN UP STACK
13349  065346  001401                                  BEQ     2$                      ;
13350  065350  104003                                  ERROR   .3                      ;BRANCH IF GOOD COMPARE
13351                                                                                  ;FPP ERROR
13352                                                                                  ;PC WAS INCORRECT
13353                                          ;
13354  065352  170203                          ;COMMON CODE FOR TRAP AND NO TRAP
13355  065354  012702  000200                  2$:     STFPS   R3                      ;SAVE FPS
13356  065360  170102                                  MOV     #200,R2                 ;SET FPP TO DOUBLE
13357  065362  012701  003102                          LDFPS   R2
13358  065366  174011                                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13359  065370  026503  000016                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13360  065374  001401                                  CMP     16(R5),R3               ;VERIFY STATUS
13361  065376  104003                                  BEQ     3$                      ;BRANCH IF GOOD
13362                                                  ERROR   .3                      ;FPP ERROR
13363  065400  010504                                                                  ;BAD FPS
13364  065402  062704  000010                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13365  065406  004767  045242                          ADD     #10,R4
13366  065412  005767  115442                  4$:     JSR     R7,DATVFR                       ;VERIFY DATA
13367  065416  001401                                  TST     COUNT
13368  065420  104003                                  BEQ     5$                      ;BRANCH IF GOOD
13369                                                  ERROR   .3                      ;FPP ERROR
13370  065422  005737  002776                                                          ;BAD AC0
13371  065426  001002                          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13372  065430  000165  000020                          BNE     6$                      ;BRANCH IF NEED TO CHECK
13373  065434  170301                                  JMP     20(R5)                  ;RETURN FROM TEST
13374  065436  016504  000020                  6$:     STST    R1                      ;SAVE FEC
13375  065442  020401                                  MOV     20(R5),R4               ;GET FEC
13376  065444  001401                                  CMP     R4,R1                   ;VERIFY FEC
13377  065446  104003                                  BEQ     7$                      ;BRANCH IF GOOD
13378                                                  ERROR   .3              ;FPP ERROR
13379  065450  000165  000022                                                          ;BAD FEC
13380                                          7$:     JMP     22(R5)                  ;RETURN FROM TEST
13381  065454                                  ;
13382                                          MOP10:
13383                                          ;
13384                                          ;
13385                                          ;-------------------------------- -
13386                                          ;TEST DIVD -
13387                                          ;
13388                                                  ;
13389                                                  ;
13390  065454                                          ;
13391                                          MDIVD:
13392
13393                                          ;
13394  065454  012737  000002  002776          ;1/AC=FSRC=0 TRAPS DISABLED
13395  065462  004767  000516                          MOV     #2,@#FLAG               ;NO INTERRUPT
13396  065466  000000  000000  000000                  JSR     R7,DVDSUB               ;DO TEST
13397  065474  000001                                  .WORD   0,0,0,1         ;AC0
13398  065476  000100  000000  000000                  .WORD   100,0,0,0       ;FSRC
13399  065504  000000
```

```
13400  065506  000000  000000  000000          .WORD   0,0,0,1          ;RESULT
13401  065514  000001
13402  065516  040000                          .WORD   40000            ; TEST FPS
13403  065520  140000                          .WORD   140000           ;RESULT FPS
13404  065522  000004                          .WORD   4                ;FEC
13405                                  ;2/FSRC=0, TRAPS ENABLED
13406  065524  012737  000001  002776          MOV     #1,@#FLAG                ;INTERRUPT
13407  065532  004767  000446                  JSR     R7,DVDSUB                ;DO TEST
13408  065536  000402  000000  000000          .WORD   402,0,0,0        ;ACO
13409  065544  000000
13410  065546  000000  000000  000000          .WORD   0,0,0,0          ;FSRC
13411  065554  000000
13412  065556  000402  000000  000000          .WORD   402,0,0,0                ;RESULT
13413  065564  000000
13414  065566  000200                          .WORD   200              ; TEST FPS
13415  065570  100200                          .WORD   100200           ;RESULT FPS
13416  065572  000004                          .WORD   4                        ;FEC
13417                                  ;3/ROUND
13418  065574  005037  002776                  CLR     @#FLAG           ;NO INTERRUPT
13419  065600  004767  000400                  JSR     R7,DVDSUB                ;DO TEST
13420  065604  034300  000000  000000          .WORD   34300,0,0,1      ;ACO
13421  065612  000001
13422  065614  140300  000000  000000          .WORD   140300,0,0,0     ;FSRC
13423  065622  000000
13424  065624  134200  000000  000000          .WORD   134200,0,0,1     ;RESULT
13425  065632  000001
13426  065634  000200                          .WORD   200              ; TEST FPS
13427  065636  000210                          .WORD   210              ;RESULT FPS
13428                                  ;4/TRUNCATE
13429  065640  005037  002776                  CLR     @#FLAG           ;NO INTERRUPT
13430  065644  004767  000334                  JSR     R7,DVDSUB                ;DO TEST
13431  065650  034300  000000  000000          .WORD   34300,0,0,1      ;ACO
13432  065656  000001
13433  065660  140300  000000  000000          .WORD   140300,0,0,0     ;FSRC
13434  065666  000000
13435  065670  134200  000000  000000          .WORD   134200,0,0,0     ;RESULT
13436  065676  000000
13437  065700  000240                          .WORD   240              ; TEST FPS
13438  065702  000250                          .WORD   250              ;RESULT FPS
13439                                  ;5/ROUND NEGATIVE AC, FSRC
13440  065704  005037  002776                  CLR     @#FLAG           ;NO INTERRUPT
13441  065710  004767  000270                  JSR     R7,DVDSUB                ;DO TEST
13442  065714  177642  000000  000000          .WORD   177642,0,0,151   ;ACO
13443  065722  000151
13444  065724  166600  000000  000000          .WORD   166600,0,0,123   ;FSRC
13445  065732  000123
13446  065734  051242  000000  000000          .WORD   51242,0,0,0              ;RESULT
13447  065742  000000
13448  065744  000200                          .WORD   200              ; TEST FPS
13449  065746  000200                          .WORD   200              ;RESULT FPS
13450                                  ;6/TRUNCATE NEAGTIVE AC, FSRC
13451  065750  005037  002776                  CLR     @#FLAG           ;NO INTERRUPT
13452  065754  004767  000224                  JSR     R7,DVDSUB                ;DO TEST
13453  065760  177642  000000  000000          .WORD   177642,0,0,151   ;ACO
13454  065766  000151
13455  065770  166600  000000  000000          .WORD   166600,0,0,123   ;FSRC
```

```
065776  000123
066000  051241  177777  177777          .WORD   51241,-1,-1,-1          ;RESULT
066006  177777
066010  000240                          .WORD   240             ; TEST FPS
066012  000240                          .WORD   240             ;RESULT FPS
                                ;7/AC=FSRC
066014  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
066020  004767  000160                  JSR     R7,DVDSUB               ;DO TEST
066024  055521  047621  100333          .WORD   55521,47621,100333,-1   ;ACO
066032  177777
066034  055521  047621  100333          .WORD   55521,47621,100333,-1   ;FSRC
066042  177777
066044  040200  000000  000000          .WORD   40200,0,0,0             ;RESULT
066052  000000
066054  007717                          .WORD   7717            ; TEST FPS
066056  007700                          .WORD   7700            ;RESULT FPS
                                ;8/UNDERFLOW TRAPS ENABLED, UV RESULT
066060  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
066066  004767  000112                  JSR     R7,DVDSUB               ;DO TEST
066072  100200  000000  000000          .WORD   100200,0,0,0            ;ACO
066100  000000
066102  077777  000000  000000          .WORD   77777,0,0,0             ;FSRC
066110  000000
066112  140400  100200  100200          .WORD   140400,100200,100200,100201     ;RESULT
066120  100201
066122  002200                          .WORD   2200            ; TEST FPS
066124  102210                          .WORD   102210          ;RESULT FPS
066126  000012                          .WORD   12              ;FEC
                                ;9/OVERFLOW TRAPS ENABLED
066130  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
066136  004767  000042                  JSR     R7,DVDSUB               ;DO TEST
066142  077000  123465  012346          .WORD   77000,123465,12346,525          ;ACO
066150  000525
066152  000303  000001  140000          .WORD   303,1,140000,140001             ;FSRC
066160  140001
066162  036650  163002  103645          .WORD   36650,163002,103645,64003       ;RESULT
066170  064003
066172  001700                          .WORD   1700            ; TEST FPS
066174  101702                          .WORD   101702          ;RESULT FPS
066176  000010                          .WORD   10              ;FEC
                                ;

066200  000167  000212                  JMP     HOP11                   ;HOP OVER SUBROUTINE
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
                                ;DIVD SUBROUTINE:
                                ;               ACO
                                ;               FSRC
                                ;               FPS BEFORE EXECUTION
                                ;               FPS AFTER EXECUTION
                                ;               (FEC)
                                ;
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
```

M3


```
13512  066204  012605                  DVDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13513  066206  012737  066262  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
13514  066214  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13515  066220  170102                          LDFPS   R2                      ;LOAD FPS
13516  066222  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13517  066224  062704  000010                  ADD     #10,R4
13518  066230  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13519  066232  016502  000030                  MOV     30(R5),R2               ;GET TEST FPS
13520  066236  170102                          LDFPS   R2                      ;LOAD TEST FPS
13521                                  ;
13522  066240  174414                          DIVD    (R4),AC0                ;*TEST INSTRUCTION
13523  066242  170000                  1$:     CFCC                            ;WAIT FOR POSSIBLE FPA TRAP.
13524                                  ;
13525                                  ;INSTRUCTION DIDNT TRAP
13526  066244  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13527  066252  001420                          BEQ     2$                      ;BRANCH IF GOOD
13528  066254  104003                          ERROR   .3                  ;FPP ERROR
13529                                          ;INSTRUCTION SHOULD HAVE TRAPPED
13530  066256  000167  000032                  JMP     2$                      ;REJOIN CODE
13531                                  ;
13532                                  ;INSTRUCTION TRAPPED
13533  066262  032737  000001  002776  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13534  066270  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13535  066272  104003                          ERROR   .3                  ;FPP ERROR
13536                                          ;INSTRUCTION WASNT SUPPOSE TO TRAP
13537  066274  000167  000014                  JMP     2$                      ;REJOIN CODE
13538  066300  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
13539  066302  005726                          TST     (SP)+                   ;CLEAN UP STACK
13540  066304  022704  066242                  CMP     #1$,R4                  ;
13541  066310  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
13542  066312  104003                          ERROR   .3                  ;FPP ERROR
13543                                          ;PC WAS INCORRECT
13544                                  ;
13545                                  ;COMMON CODE FOR TRAP AND NO TRAP
13546  066314  170203                  2$:     STFPS   R3                              ;SAVE FPS
13547  066316  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
13548  066322  170102                          LDFPS   R2
13549  066324  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13550  066330  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13551  066332  026503  000032                  CMP     32(R5),R3               ;VERIFY STATUS
13552  066336  001401                          BEQ     3$                      ;BRANCH IF GOOD
13553  066340  104003                          ERROR   .3                  ;FPP ERROR
13554                                          ;BAD FPS
13555  066342  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13556  066344  062704  000020                  ADD     #20,R4
13557  066350  004767  044316          4$:     JSR     R7,DATVER                       ;VERIFY DATA
13558  066354  005767  114500                  TST     COUNT
13559  066360  001401                          BEQ     5$                      ;BRANCH IF GOOD
13560  066362  104003                          ERROR   .3                  ;FPP ERROR
13561                                          ;BAD AC0
13562  066364  005737  002776  5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13563  066370  001002                          BNE     6$                      ;BRANCH IF NEED TO CHECK
13564  066372  000165  000034                  JMP     34(R5)                  ;RETURN FROM TEST
13565  066376  170301                  6$:     STST    R1                      ;SAVE FEC
13566  066400  016504  000034                  MOV     34(R5),R4               ;GET FEC
13567  066404  020401                          CMP     R4,R1                   ;VERIFY FEC
```

```
13568  066406  001401                  BEQ     7$                      ;BRANCH IF GOOD
13569  066410  104003                  ERROR   .+3             ;FPP ERROR
13570                                                                   ;BAD FEC
13571  066412  000165  000036    7$:   JMP     36(R5)                  ;RETURN FROM TEST
13572                              ;
13573  066416                      HOP11:
13574                              ;
13575                              ;
13576                              ;---------- --------------------
13577                              ;TEST MULF
13578                              ;
13579                                      ;
13580                                      ;
13581                                      ;
13582  066416                      MMULF:
13583
13584                              ;
13585                              ;1/AC0=FSRC=0 -INTERRUPTS DISABLED
13586  066416  005037  002776              CLR     @#FLAG          ;NO INTERRUPT
13587  066422  004767  000564              JSR     R7,MLFSUB               ;DO TEST
13588  066426  000000  000000              .WORD   0.0     ;AC0
13589  066432  000000  000000              .WORD   0.0     ;FSRC
13590  066436  000000  000000              .WORD   0.0             ;RESULT
13591  066442  007517                      .WORD   7517            ; TEST FPS
13592  066444  007504                      .WORD   7504            ;RESULTANT FPS
13593                              ;2/AC>FSRC=0 - INTERRUPTS ON
13594  066446  005037  002776              CLR     @#FLAG          ;NO INTERRUPT
13595  066452  004767  000534              JSR     R7,MLFSUB               ;DO TEST
13596  066456  000200  000000              .WORD   200.0   ;AC0
13597  066462  000000  000000              .WORD   0.0     ;FSRC
13598  066466  000000  000000              .WORD   0.0             ;RESULT
13599  066472  000013                      .WORD   13              ; TEST FPS
13600  066474  000004                      .WORD   4               ;RESULTANT FPS
13601                              ;3/AC=0 FSRC>0 -
13602  066476  005037  002776              CLR     @#FLAG          ;NO INTERRUPT
13603  066502  004767  000504              JSR     R7,MLFSUB               ;DO TEST
13604  066506  000100  000000              .WORD   100.0   ;AC0
13605  066512  000300  000000              .WORD   300.0   ;FSRC
13606  066516  000000  000000              .WORD   0.0             ;RESULT
13607  066522  007500                      .WORD   7500            ; TEST FPS
13608  066524  007504                      .WORD   7504            ;RESULTANT FPS
13609                              ;4/AC=1 >FSRC - ROUND
13610  066526  005037  002776              CLR     @#FLAG          ;NO INTERRUPT
13611  066532  004767  000454              JSR     R7,MLFSUB               ;DO TEST
13612  066536  040200  000000              .WORD   40200.0 ;AC0
13613  066542  040177  177777              .WORD   40177,-1        ;FSRC
13614  066546  040177  177777              .WORD   40177,-1                ;RESULT
13615  066552  000000                      .WORD   0               ; TEST FPS
13616  066554  000000                       WORD   0               ;RESULTANT FPS
13617                              ;5/TRUNCATE
13618  066556  005037  002776              CLR     @#FLAG          ;NO INTERRUPT
13619  066562  004767  000424              JSR     R7,MLFSUB               ;DO TEST
13620  066566  040177  177777              .WORD   40177,-1        ;AC0
13621  066572  040200  000000              .WORD   40200.0 ;FSRC
13622  066576  040177  177777              .WORD   40177,-1                ;RESULT
13623  066602  000040                      .WORD   40              ; TEST FPS
```

```
13624  066604  000040                          .WORD   40              ;RESULTANT FPS
13625                                  ;6/NORMALIZE
13626  066606  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13627  066612  004767  000374                  JSR     R7,MLFSUB               ;DO TEST
13628  066616  040100  000000                  .WORD   40100,0 ;AC0
13629  066622  040100  000000                  .WORD   40100,0 ;FSRC
13630  066626  040020  000000                  .WORD   40020,0         ;RESULT
13631  066632  000012                          .WORD   12              ; TEST FPS
13632  066634  000000                          .WORD   0               ;RESULTANT FPS
13633                                  ;7/ROUND
13634  066636  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13635  066642  004767  000344                  JSR     R7,MLFSUB               ;DO TEST
13636  066646  017500  000000                  .WORD   17500,0 ;AC0
13637  066652  023652  125252                  .WORD   23652,125252    ;FSRC
13638  066656  003177  177777                  .WORD   3177,-1         ;RESULT
13639  066662  007417                          .WORD   7417            ; TEST FPS
13640  066664  007400                          .WORD   7400            ;RESULTANT FPS
13641                                  ;8/AC>0>FSRC ROUND
13642  066666  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13643  066672  004767  000314                  JSR     R7,MLFSUB               ;DO TEST
13644  066676  040342  177777                  .WORD   40342,-1        ;AC0
13645  066702  176543  025252                  .WORD   176543,025252   ;FSRC
13646  066706  176711  067324                  .WORD   176711,67324            ;RESULT
13647  066712  007500                          .WORD   7500            ; TEST FPS
13648  066714  007510                          .WORD   7510            ;RESULTANT FPS
13649                                  ;9/IAC<FSRC<0, ROUND
13650  066716  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13651  066722  004767  000264                  JSR     R7,MLFSUB               ;DO TEST
13652  066726  144600  000000                  .WORD   144600,0                ;AC0
13653  066732  154000  000000                  .WORD   154000,0                ;FSRC
13654  066736  060400  000000                  .WORD   60400,0         ;RESULT
13655  066742  000017                          .WORD   17              ; TEST FPS
13656  066744  000000                          .WORD   0               ;RESULT FPS
13657                                  ;10/AC<FSRC, ROUND
13658  066746  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13659  066752  004767  000234                  JSR     R7,MLFSUB               ;DO TEST
13660  066756  060000  000000                  .WORD   60000,0         ;AC0
13661  066762  140377  177776                  .WORD   140377,177776           ;FSRC
13662  066766  160177  177776                  .WORD   160177,177776           ;RESULT
13663  066772  000017                          .WORD   17              ; TEST FPS
13664  066774  000010                          .WORD   10              ;RESULT FPS
13665                                  ;11/AC>0>FSRC, TRUNCATE
13666  066776  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13667  067002  004767  000204                  JSR     R7,MLFSUB               ;DO TEST
13668  067006  060000  000000                  .WORD   60000,0         ;AC0
13669  067012  140377  177776                  .WORD   140377,177776           ;FSRC
13670  067016  160177  177776                  .WORD   160177,177776           ;RESULT
13671  067022  007547                          .WORD   7547            ; TEST FPS
13672  067024  007550                          .WORD   7550            ;RESULT FPS
13673                                  ;12/UNDERFLOW, NO INTERRUPTS
13674  067026  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPT
13675  067034  004767  000152                  JSR     R7,MLFSUB               ;DO TEST
13676  067040  000200  000001                  .WORD   200,1           ;AC0
13677  067044  000200  000001                  .WORD   200,1           ;FSRC
13678  067050  040200  000002                  .WORD   40200,2         ;RESULT
13679  067054  042117                          .WORD   42117           ; TEST FPS
```

```
13680  067056  142100                          .WORD   142100          ;RESULT FPS
13681  067060  000012                          .WORD   12              ;FEC
13682                                  ;13/OVERFLOW, TRAP
13683  067062  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
13684  067070  004767  000116                  JSR     R7,MLFSUB               ;DO TEST
13685  067074  177777  177777                  .WORD   177777, 1               ;ACO
13686  067100  040300  000000                  .WORD   40300,0         ;FSRC
13687  067104  100077  177777                  .WORD   100077,-1               ;RESULT
13688  067110  001117                          .WORD   1117            ; TEST FPS
13689  067112  101116                          .WORD   101116          ;RESULT FPS
13690  067114  000010                          .WORD   10              ;FEC
13691                                  ;14/OVERFLOW NO TRAP
13692  067116  012737  00 002  002776          MOV     #2,@#FLAG               ;NO INTERRUPT
13693  067124  004767  000062                  JSR     R7,MLFSUB               ;DO TEST
13694  067130  077700  000000                  .WORD   77700,0         ;ACO
13695  067134  077700  000000                  .WORD   77700,0         ;FSRC
13696  067140  000000  000000                  .WORD   0,0             ;RESULT
13697  067144  040117                          .WORD   40117           ; TEST FPS
13698  067146  040106                          .WORD   40106           ;RESULT FPS
13699  067150  000010                          .WORD   10              ;FEC
13700                                  ;15/UNDEFINED VARIABLE IN FSRC, TRAP ENABLED
13701  067152  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
13702  067160  004767  000026                  JSR     R7,MLFSUB               ;DO TEST
13703  067164  123465  000000                  .WORD   123465,0                ;ACO
13704  067170  100022  000000                  .WORD   100022,0                ;FSRC
13705  067174  123465  000000                  .WORD   123465,0                ;RESULT
13706  067200  004000                          .WORD   4000            ; TEST FPS
13707  067202  104000                          .WORD   104000          ;RESULT FPS
13708  067204  000014                          .WORD   14              ;FEC
13709                                  ;
13710
13711  067206  000167  000212                  JMP     MOP12
13712                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13713                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13714                                  ;
13715                                  ;               ACO
13716                                  ;               FSRC
13717                                  ;               FPS BEFORE EXECUTION
13718                                  ;               FPS AFTER EXECUTION
13719                                  ;               (FEC)
13720                                  ;
13721                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13722                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13723                                  ;
13724  067212  012605                  MLFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13725  067214  012737  067270  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
13726  067222  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13727  067226  170102                          LDFPS   R2                      ;LOAD FPS
13728  067230  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13729  067232  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13730  067234  062704  000004                  ADD     #4,R4
13731  067240  016502  000014                  MOV     14(R5),R2               ;GET TEST FPS
13732  067244  170102                          LDFPS   R2                      ;LOAD TEST FPS
13733                                  ;
13734  067246  171014                          MULF    (R4),AC0                ;*TEST INSTRUCTION
13735  067250  170001                  10$:    SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
```

```
13736                                          ;
13737                                          ;INSTRUCTION DIDNT TRAP
13738  067252  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13739  067260  001420                          BEQ     2$                      ;BRANCH IF GOOD
13740  067262  104003                          ERROR   .3              ;FPP ERROR
13741                                          ;INSTRUCTION SHOULD HAVE TRAPPED
13742  067264  000167  000032                  JMP     2$                      ;REJOIN CODE
13743                                          ;
13744                                          ;INSTRUCTION TRAPPED
13745  067270  032737  000001  002776  50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13746  067276  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13747  067300  104003                          ERROR   .3              ;FPP ERROR
13748                                          ;INSTRUCTION WASNT SUPPOSE TO TRAP
13749  067302  00C167  000014                  JMP     2$                      ;REJOIN CODE
13750  067306  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
13751  067310  005726                          TST     (SP)+                   ;CLEAN UP STACK
13752  067312  022704  067250                  CMP     #1$,R4                  ;
13753  067316  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
13754  067320  104003                          ERROR   .3              ;FPP ERROR
13755                                          ;PC WAS INCORRECT
13756                                          ;
13757                                          ;COMMON CODE FOR TRAP AND NO TRAP
13758  067322  170203                  2$:     STFPS   R3                      ;SAVE FPS
13759  067324  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
13760  067330  170102                          LDFPS   R2
13761  067332  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13762  067336  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13763  067340  026503  000016                  CMP     16(R5),R3               ;VERIFY STATUS
13764  067344  001401                          BEQ     3$                      ;BRANCH IF GOOD
13765  067346  104003                          ERROR   .3              ;FPP ERROR
13766                                          ;BAD FPS
13767  067350  010504                  3$:     MOV     R5,R4                           ;POINT TO EXPECTED DATA
13768  067352  062704  000010                  ADD     #10,R4
13769  067356  004767  043272          4$:     JSR     R7,DATVFR                               ;VERIFY DATA
13770  067362  005767  113472                  TST     COUNT
13771  067366  001401                          BEQ     5$                      ;BRANCH IF GOOD
13772  067370  104003                          ERROR   .3              ;FPP ERROR
13773                                          ;BAD AC0
13774  067372  005737  002776          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13775  067376  001002                          BNE     6$                      ;BRANCH IF NEED TO CHECK
13776  067400  000165  000020                  JMP     20(R5)                  ;RETURN FROM TEST
13777                                          ;VERIFY ERROR STATUS
13778  067404  170301                  6$:     STST    R1                      ;SAVE FEC
13779  067406  016504  000020                  MOV     20(R5),R4               ;GET FEC
13780  067412  020401                          CMP     R4,R1                   ;VERIFY FEC
13781  067414  001401                          BEQ     7$                      ;BRANCH IF GOOD
13782  067416  104003                          ERROR   .3              ;FPP ERROR
13783                                          ;BAD FEC
13784  067420  000165  000022          7$:     JMP     22(R5)          ;RETURN FROM TEST
13785  067424                          HOP12:
13786                                          ;
13787                                          ;
13788                                          ;--------------------------------
13789                                          ;TEST MULD
13790                                          ;
13791                                                  ;
```

E4


```
13792                                          ;
13793                                          ;
13794  067424                          MMULD:
13795
13796                                  ;
13797                                  ;1/AC=0
13798  067424  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13799  067430  004767  000554                  JSR     R7,MLDSUB               ;DO TEST
13800  067434  000100  000000  000000          .WORD   100,0,0,0       ;AC0
13801  067442  000000
13802  067444  000411  177777  000000          .WORD   411,-1,0,1      ;FSRC
13803  067452  000001
13804  067454  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
13805  067462  000000
13806  067464  000200                          .WORD   200             ; TEST FPS
13807  067466  000204                          .WORD   204             ;RESULTANT FPS
13808                                  ;2/FSRC=0
13809  067470  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13810  067474  004767  000510                  JSR     R7,MLDSUB               ;DO TEST
13811  067500  077777  000000  000000          .WORD   77777,0,0,0     ;AC0
13812  067506  000000
13813  067510  000000  000000  000000          .WORD   0,0,0,0 ;FSRC
13814  067516  000000
13815  067520  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
13816  067526  000000
13817  067530  007700                          .WORD   7700            ; TEST FPS
13818  067532  007704                          .WORD   7704            ;RESULTANT FPS
13819                                  ;3/AC=1
13820  067534  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13821  067540  004767  000444                  JSR     R7,MLDSUB               ;DO TEST
13822  067544  040200  000000  000000          .WORD   40200,0,0,0     ;AC0
13823  067552  000000
13824  067554  000277  177777  177777          .WORD   277,-1,-1,-1    ;FSRC
13825  067562  177777
13826  067564  000277  177777  177777          .WORD   277,-1,-1,-1            ;RESULT
13827  067572  177777
13828  067574  007717                          .WORD   7717            ; TEST FPS
13829  067576  007700                          .WORD   7700            ;RESULTANT FPS
13830                                  ;4/AC>FSRC>0, TRUNCATE
13831  067600  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13832  067604  004767  000400                  JSR     R7,MLDSUB               ;DO TEST
13833  067610  065500  000000  000000          .WORD   65500,0,0,1     ;AC0
13834  067616  000001
13835  067620  037577  177777  177777          .WORD   37577,-1,-1,-2  ;FSRC
13836  067626  177776
13837  067630  065077  177777  177777          .WORD   65077,-1,-1,-1          ;RESULT
13838  067636  177777
13839  067640  007717                          .WORD   7717            ; TEST FPS
13840  067642  007700                          .WORD   7700            ;RESULTANT FPS
13841                                  ;5/AC<FSRC<0
13842  067644  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13843  067650  004767  000334                  JSR     R7,MLDSUB               ;DO TEST
13844  067654  137577  177777  177777          .WORD   137577,-1,-1,-2 ;AC0
13845  067662  177776
13846  067664  165400  000000  000000          .WORD   165400,0,0,1    ;FSRC
13847  067672  000001
```

```
13848  067674  065000  000000  000000          .WORD   65000,0,0,0             ;RESULT
13849  067702  000000
13850  067704  007717                          .WORD   7717            ; TEST FPS
13851  067706  007700                          .WORD   7700            ;RESULTANT FPS
13852                                  ;6/AC>FSRC>0
13853  067710  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
13854  067714  004767  000270                  JSR     R7,MLDSUB               ;DO TEST
13855  067720  017500  000000  000000          .WORD   17500,0,0,0     ;AC0
13856  067726  000000
13857  067730  123652  125252  125252          .WORD   123652,125252,125252,125252     ;FSRC
13858  067736  125252
13859  067740  103177  177777  177777          .WORD   103177,-1,-1,-1         ;RESULT
13860  067746  177777
13861  067750  000200                          .WORD   200             ; TEST FPS
13862  067752  000210                          .WORD   210             ;RESULTANT FPS
13863                                  ;7/UNDERFLOW, TRAPS DISABLED
13864  067754  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPT
13865  067760  004767  000224                  JSR     R7,MLDSUB               ;DO TEST
13866  067764  000300  000000  000000          .WORD   300,0,0,252             ;AC0
13867  067772  000252
13868  067774  000377  000001  000002          .WORD   377,1,2,3               ;FSRC
13869  070002  000003
13870  070004  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
13871  070012  000000
13872  070014  005740                          .WORD   5740            ; TEST FPS
13873  070016  005744                          .WORD   5744            ;RESULT FPS
13874                                  ;8/UNDFRFLOW, TRAP ENABLED
13875  070020  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
13876  070026  004767  000156                  JSR     R7,MLDSUB               ;DO TEST
13877  070032  100277  000001  000002          .WORD   100277,1,2,-1           ;AC0
13878  070040  177777
13879  070042  100300  000001  000001          .WORD   100300,1,1,1            ;FSRC
13880  070050  000001
13881  070052  040417  040001  077403          .WORD   40417,40001,77403,0             ;RESULT
13882  070060  000000
13883  070062  002217                          .WORD   2217            ; TEST FPS
13884  070064  102200                          .WORD   102200          ;RESULT FPS
13885  070066  000012                          .WORD   12              ;FEC
13886                                  ;9/OVERFLOW, TRAPS DISABLED
13887  070070  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPT
13888  070074  004767  000110                  JSR     R7,MLDSUB               ;DO TEST
13889  070100  177777  177777  177777          .WORD   -1,-1,-1,-1             ;AC0
13890  070106  177777
13891  070110  040200  177777  177777          .WORD   40200,-1,-1,-1          ;FSRC
13892  070116  177777
13893  070120  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
13894  070126  000000
13895  070130  006740                          .WORD   6740            ; TEST FPS
13896  070132  006746                          .WORD   6746            ;RESULT FPS
13897                                  ;10/OVERFLOW, TRAPS ENABLED
13898  070134  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
13899  070142  004767  000042                  JSR     R7,MLDSUB               ;DO TEST
13900  070146  157700  025252  025252          .WORD   157700,25252,25252,25252                ;AC0
13901  070154  025252
13902  070156  167700  000000  000000          .WORD   167700,0,0,0            ;FSRC
13903  070164  000000
```

```
13904  070166  007420  017777  117777          .WORD   7420,017777,117777,117777              ;RESULT
13905  070174  117777
13906  070176  001240                          .WORD   1240            ; TEST FPS
13907  070200  101242                          .WORD   101242          ;RESULT FPS
13908  070202  000010                          .WORD   10              ;FEC
13909                                  ;
13910
13911  070204  000167  000212                  JMP     MOP13
13912
13913                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13914                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13915                                  ;
13916                                  ;               AC0
13917                                  ;               FSRC
13918                                  ;               FPS BEFORE EXECUTION
13919                                  ;               FPS AFTER EXECUTION
13920                                  ;               (FEC)
13921                                  ;
13922                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13923                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13924                                  ;
13925  070210  012605                  MLDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13926  070212  012737  070266  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
13927  070220  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13928  070224  170102                          LDFPS   R2                      ;LOAD FPS
13929  070226  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13930  070230  010501                          MOV     R5,R1                   ;POINT TO FSRC DATA
13931  070232  062701  000010                  ADD     #10,R1
13932  070236  016502  000030                  MOV     30(R5),R2               ;GET TEST FPS
13933  070242  170102                          LDFPS   R2                      ;LOAD TEST FPS
13934                                  ;
13935  070244  171011                          MULD    (R1),AC0                ;*TEST INSTRUCTION
13936  070246  170011                  1$:     SETD                            ;WAIT FOR POSSIBLE FPA TRAP.
13937                                  ;
13938                                  ;INSTRUCTION DIDNT TRAP
13939  070250  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13940  070256  001420                          BEQ     2$                      ;BRANCH IF GOOD
13941  070260  104003                          ERROR   .3              ;FPP ERROR
13942                                                                  ;INSTRUCTION SHOULD HAVE TRAPPED
13943  070252  000167  000032                  JMP     2$                      ;REJOIN CODE
13944
13945                                  ;INSTRUCTION TRAPPED
13946  070266  032737  000001  002776  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13947  070274  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13948  070276  104003                          ERROR   .3              ;FPP ERROR
13949                                                                  ;INSTRUCTION WASNT SUPPOSE TO TRAP
13950  070300  000167  000014                  JMP     2$                      ;REJOIN CODE
13951  070304  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
13952  070306  005726                          TST     (SP)+                   ;CLEAN UP STACK
13953  070310  022704  070246                  CMP     #1$,R4                  ;
13954  070314  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
13955  070316  104003                          ERROR   .3              ;FPP ERROR
13956                                                                  ;PC WAS INCORRECT
13957                                  ;
13958                                  ;COMMON CODE FOR TRAP AND NO TRAP
13959  070320  170203                  2$:     STFPS   R3                      ;SAVE FPS
```

```
13960  070322  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
13961  070326  170102                          LDFPS   R2
13962  070330  012701  003102                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13963  070334  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13964  070336  026503  000032                  CMP     32(R5),R3               ;VERIFY STATUS
13965  070342  001401                          BEQ     3$                      ;BRANCH IF GOOD
13966  070344  104003                          ERROR   .3              ;FPP ERROR
13967                                                                  ;BAD FPS
13968  070346  010504          3$:             MOV     R5,R4                   ;POINT TO EXPECTED DATA
13969  070350  062704  000020                  ADD     #20,R4
13970  070354  004767  042312          4$:     JSR     R7,DATVER                       ;VERIFY DATA
13971  070360  005767  112474                  TST     COUNT
13972  070364  001401                          BEQ     5$                      ;BRANCH IF GOOD
13973  070366  104003                          ERROR   .3              ;FPP ERROR
13974                                                                  ;BAD AC0
13975  070370  005737  002776          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13976  070374  001002                          BNE     6$                      ;BRANCH IF NEED TO CHECK
13977  070376  000165  000034                  JMP     34(R5)                  ;RETURN FROM TEST
13978                                  ;VERIFY ERROR STATUS
13979  070402  170301                  6$:     STST    R1                      ;SAVE FEC
13980  070404  016504  000034                  MOV     34(R5),R4               ;GET FEC
13981  070410  020401                          CMP     R4,R1                   ;VERIFY FEC
13982  070412  001401                          BEQ     7$                      ;BRANCH IF GOOD
13983  070414  104003                          ERROR   .3              ;FPP ERROR
13984                                                                  ;BAD FEC
13985  070416  000165  000036          7$:     JMP     36(R5)                  ;RETURN FROM TEST
13986  070422                          HOP13:
13987                                  ;
13988                                  ;
13989                                  ;-------------------------------
13990                                  ;TEST MODF
13991                                  ;
13992                                          ;
13993                                          ;
13994                                          ;
13995  070422                          MMODF:
13996
13997                                  ;
13998                                  ;1/AC=0 FSRC=0
13999  070422  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPT
14000  070426  004767  000554                  JSR     R7,MDFSUB                       ;DO TEST
14001  070432  000100  000000                  .WORD   100,0   ;AC0
14002  070436  012346  177777                  .WORD   12346,-1                ;FSRC
14003  070442  000000  000000                  .WORD   0,0                     ;FRACTIONAL RESULT
14004  070446  000000  000000                  .WORD   0,0                     ;INTEGER RESULT
14005  070452  000013                          .WORD   13                      ; TEST FPS
14006  070454  000004                          .WORD   4                       ;RESULTANT FPS
14007                                  ;2/FSRC=0
14008  070456  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPT
14009  070462  004767  000520                  JSR     R7,MDFSUB                       ;DO TEST
14010  070466  012356  177777                  .WORD   12356,-1                ;AC0
14011  070472  000000  000000                  .WORD   0,0     ;FSRC
14012  070476  000000  000000                  .WORD   0,0                     ;FRACTIONAL RESULT
14013  070502  000000  000000                  .WORD   0,0                     ;INTEGER RESULT
14014  070506  000003                          .WORD   3                       ; TEST FPS
14015  070510  000004                          .WORD   4       ;RESULTANT FPS
```

```
 14016
 14017  070512  005037  002776          ;3/AC=0
 14018  070516  004767  000464                  CLR     @#FLAG          ;NO INTERRUPT
 14019  070522  000000  000000                  JSR     R7,MDFSUB               ;DO TEST
 14020  070526  177777  177777                  .WORD   0,0             ;ACO
 14021  070532  000000  000000                  .WORD   -1,-1           ;FSRC
 14022  070536  000000  000000                  .WORD   0,0             ;FRACTIONAL  RESULT
 14023  070542  007500                          .WORD   0,0             ;INTEGER RESULT
 14024  070544  007504                          .WORD   7500            ; TEST FPS
 14025                                          .WORD   7504            ;RESULT FPS
 14026  070546  005037  002776          ;4/AC>FSRC>0
 14027  070552  004767  000430                  CLR     @#FLAG          ;NO INTERRUPT
 14028  070556  046252  125252                  JSR     R7,MDFSUB               ;DO TEST
 14029  070562  040300  000000                  .WORD   46252,125252    ;ACO
 14030  070566  000000  000000                  .WORD   40300,0 ;FSRC
 14031  070572  046377  177777                  .WORD   0,0             ;FRACTIONAL RESULT
 14032  070576  000013                          .WORD   46377,-1                ;INTEGER RESULT
 14033  070600  000004                          .WORD   13              ; TEST FPS
 14034                                          .WORD   4               ;RESULTANT FPS
 14035  070602  005037  002776          ;5/AC>FSRC>0
 14036  070606  004767  000374                  CLR     @#FLAG          ;NO INTERRUPT
 14037  070612  077652  125252                  JSR     R7,MDFSUB               ;DO TEST
 14038  070616  040300  000000                  .WORD   77652,125252    ;ACO
 14039  070622  000000  000000                  .WORD   40300,0 ;FSRC
 14040  070626  077777  177777                  .WORD   0,0             ;FRACTIONAL RESULT
 14041  070632  000000                          .WORD   77777,-1                ;INTEGER RESULT
 14042  070634  000004                          .WORD   0               ; TEST FPS
 14043                                          .WORD   4               ;RESULTANT FPS
 14044
 14045  070636  005037  002776          ;6/AC>0<FSRC, INTEGERS
 14046  070642  004767  000340                  CLR     @#FLAG          ;NO INTERRUPT
 14047  070646  060600  000000                  JSR     R7,MDFSUB               ;DO TEST
 14048  070652  147400  025700                  .WORD   60600,0         ;ACO
 14049  070656  000000  000000                  .WORD   147400,25700            ;FSRC
 14050  070662  170000  025700                  .WORD   0,0             ;FRACTIONAL  RESULT
 14051  070666  007400                          .WORD   170000,25700            ;INTEGER RESULT
 14052  070670  007404                          .WORD   7400            ; TEST FPS
 14053                                          .WORD   7404            ;RESULT FPS
 14054  070672  005037  002776          ;7/AC<0<FSRC, FRACTIONAL
 14055  070676  004767  000304                  CLR     @#FLAG          ;NO INTERRUPT
 14056  070702  100227  177777                  JSR     R7,MDFSUB               ;DO TEST
 14057  070706  044025  025252                  .WORD   100227,-1               ;ACO
 14058  070712  104061  021251                  .WORD   44025,25252             ;FSRC
 14059  070716  000000  000000                  .WORD   104061,21251            ;FRACTIONAL  RESULT
 14060  070722  000000                          .WORD   0,0             ;INTEGER RESULT
 14061  070724  000010                          .WORD   0               ; TEST FPS
 14062                                          .WORD   10              ;RESULT FPS
 14063  070726  005037  002776          ;8/AC<0>FSRC, TRUNCATE
 14064  070732  004767  000250                  CLR     @#FLAG          ;NO INTERRUPT
 14065  070736  046252  125252                  JSR     R7,MDFSUB               ;DO TEST
 14066  070742  040300  000000                  .WORD   46252,125252            ;ACO
 14067  070746  000000  000000                  .WORD   40300,0         ;FSRC
 14068  070752  046377  177777                  .WORD   0,0             ;FRACTIONAL  RESULT
 14069  070756  000053                          .WORD   46377,-1                ;INTEGER RESULT
 14070  070760  000044                          .WORD   53              ; TEST FPS
 14071                                          .WORD   44              ;RESULT FPS
                                        ;9/ROUND INTEGER
```

```
 14072  070762  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 14073  070766  004767  000214                 JSR     R7,MDFSUB               ;DO TEST
 14074  070772  046252  125252                 .WORD   46252,125252            ;ACO
 14075  070776  040300  000000                 .WORD   40300,0         ;FSRC
 14076  071002  000000  000000                 .WORD   0,0             ;FRACTIONAL  RESULT
 14077  071006  046377  177777                 .WORD   46377,-1                ;INTEGER RESULT
 14078  071012  000013                         .WORD   13              ; TEST FPS
 14079  071014  000004                         .WORD   4               ;RESULT FPS
 14080                          ;10/TRUNCATE FRACTION
 14081  071016  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 14082  071022  004767  000160                 JSR     R7,MDFSUB               ;DO TEST
 14083  071026  040777  177777                 .WORD   40777,-1                ;ACO
 14084  071032  040200  000000                 .WORD   40200,0         ;FSRC
 14085  071036  040177  177770                 .WORD   40177,177770            ;FRACTIONAL  RESULT
 14086  071042  040740  000000                 .WORD   40740,0         ;INTEGER RESULT
 14087  071046  000000                         .WORD   0               ; TEST FPS
 14088  071050  000000                         .WORD   0               ;RESULT FPS
 14089                          ;11/ROUND INTEGER
 14090  071052  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 14091  071056  004767  000124                 JSR     R7,MDFSUB               ;DO TEST
 14092  071062  000000  000000                 .WORD   0,0             ;ACO
 14093  071066  000000  000000                 .WORD   0,0             ;FSRC
 14094  071072  000000  000000                 .WORD   0,0             ;FRACTIONAL  RESULT
 14095  071076  000000  000000                 .WORD   0,0             ;INTEGER RESULT
 14096  071102  000000                         .WORD   0               ; TEST FPS
 14097  071104  000004                         .WORD   4               ;RESULT FPS
 14098                          ;12/ROUND FRACTION
 14099  071106  005037  002776                 CLR     @#FLAG          ;NO INTERRUPT
 14100  071112  004767  000070                 JSR     R7,MDFSUB               ;DO TEST
 14101  071116  040225  125252                 .WORD   40225,125252            ;ACO
 14102  071122  066652  052525                 .WORD   66652,52525             ;FSRC
 14103  071126  000000  000000                 .WORD   0,0             ;FRACTIONAL  RESULT
 14104  071132  066707  025160                 .WORD   66707,25160             ;INTEGER RESULT
 14105  071136  007027                         .WORD   7027            ; TEST FPS
 14106  071140  007004                         .WORD   7004            ;RESULT FPS
 14107                          ;/OVERFLOW
 14108  071142  012737  000001  002776         MOV     #1,@#FLAG               ;INTERRUPT
 14109  071150  004767  000032                 JSR     R7,MDFSUB               ;DO TEST
 14110  071154  076000  000000                 .WORD   76000,0         ;ACO
 14111  071160  076000  000000                 .WORD   76000,0         ;FSRC
 14112  071164  000000  000000                 .WORD   0,0             ;FRACTIONAL  RESULT
 14113  071170  033600  000000                 .WORD   33600,0         ;INTEGER RESULT
 14114  071174  001000                         .WORD   1000            ; TEST FPS
 14115  071176  101006                         .WORD   101006          ;RESULT FPS
 14116  071200  000010                         .WORD   10              ;FEC
 14117                          ;
 14118  071202  000167  000254                 JMP     @OP14
 14119
 14120                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 14121                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 14122                          ;
 14123                          ;               ACO
 14124                          ;               FSRC
 14125                          ;               FRACTIONAL RESULT
 14126                          ;               INTEGER RESULT
 14127                          ;               FPS BEFORE EXECUTION
```

```
                                        ;               FPS AFTER EXECUTION
                                        ;               (FEC)
                                        ;
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;
071206  012605                          MOFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
071210  012737  071272  000244                  MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
071216  012702  000200                          MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
071222  170102                                  LDFPS   R2                      ;LOAD FPS
071224  172415                                  LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
071226  012701  071452                          MOV     #MODGAR,R1              ;LOAD KNOWN INTO AC1
071232  172511                                  LDD     (R1),AC1                ;
071234  010501                                  MOV     R5,R1                   ;POINT TO FSRC DATA
071236  062701  000004                          ADD     #4,R1
071242  016502  000020                          MOV     20(R5),R2               ;GET TEST FPS
071246  170102                                  LDFPS   R2                      ;LOAD TEST FPS
                                        ;
071250  171411                                  MODF    (R1),AC0                ;*TEST INSTRUCTION
071252  170001                          1$:     SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
                                        ;
                                        ;INSTRUCTION DIDNT TRAP
071254  032737  000001  002776                  BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
071262  001420                                  BEQ     2$                      ;BRANCH IF GOOD
071264  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;INSTRUCTION SHOULD HAVE TRAPPED
071266  000167  000032                          JMP     2$                      ;REJOIN CODE
                                        ;
                                        ;INSTRUCTION TRAPPED
071272  032737  000001  002776          50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
071300  001003                                  BNE     51$                     ;BRANCH IF EXPECTING A TRAP
071302  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;INSTRUCTION WASNT SUPPOSE TO TRAP
071304  000167  000014                          JMP     2$                      ;REJOIN CODE
071310  012604                          51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
071312  005726                                  TST     (SP)+                   ;CLEAN UP STACK
071314  022704  071252                          CMP     #1$,R4                  ;
071320  001401                                  BEQ     2$                      ;BRANCH IF GOOD COMPARE
071322  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;PC WAS INCORRECT
                                        ;
                                        ;COMMON CODE FOR TRAP AND NO TRAP
071324  170203                          2$:     STFPS   R3                      ;SAVE FPS
071326  012702  000200                          MOV     #200,R2                 ;SET FPP TO DOUBLE
071332  170102                                  LDFPS   R2
071334  012701  003102                          MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
                                        ;SAVE FRACTIONAL RESULT
071340  174011                                  STD     AC0,(R1)                ;SAVE AC0  RESULT
071342  026503  000022                          CMP     22(R5),R3               ;VERIFY STATUS
071346  001401                                  BEQ     3$                      ;BRANCH IF GOOD
071350  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;BAD FPS
071352  010504                          3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
071354  062704  000010                          ADD     #10,R4
071360  004767  041270                  4$:     JSR     R7,DATVFR                       ;VERIFY DATA
071364  005767  111470                          TST     COUNT
```

```
14184  071370  001401                          BEQ     5$                      ;BRANCH IF GOOD
14185  071372  104003                          ERROR   .+3                   ;FPP ERROR
14186                                                                         ;BAD AC0
14187                                          ;SAVE INTEGER RESULT
14188  071374  174111                  5$:     STD     AC1,(R1)                ;SAVE AC1 RESULT
14189  071376  010504                          MOV     R5,R4                 ;POINT TO EXPECTED
14190  071400  062704  000014                  ADD     #14,R4
14191  071404  004767  041244                  JSR     R7,DATVFR               ;VERIFY DATA
14192  071410  005767  111444                  TST     COUNT                   ;
14193  071414  001401                          BEQ     6$                      ;BRANCH IF GOOD
14194  071416  104003                          ERROR   .+3                   ;FPP ERROR
14195                                                                         ;BAD AC1
14196  071420  005737  002776          6$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14197  071424  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
14198  071426  000165  000024                  JMP     24(R5)                  ;RETURN FROM TEST
14199  071432  170301                  7$:     STST    R1                      ;SAVE FEC
14200  071434  016504  000024                  MOV     24(R5),R4               ;GET FEC
14201  071440  020401                          CMP     R4,R1                   ;VERIFY FEC
14202  071442  001401                          BEQ     8$                      ;BRANCH IF GOOD
14203  071444  104003                          ERROR   .+3                   ;FPP ERROR
14204                                                                         ;BAD FEC
14205  071446  000165  000026          8$:     JMP     26(R5)                ;RETURN FROM TEST
14206                                          ;
14207  071452  177777  177777  177777  MODGAR: .WORD   -1,-1,-1,-1             ;KNOWN DATA FOR AC1
14208  071460  177777
14209  071462                          MOP14:
14210
14211                                          ;
14212                                          ;
14213                                          ;---------------------------------
14214                                          ;TEST MODD
14215                                          ;
14216                                                  ;
14217                                                  ;
14218                                                  ;
14219  071462                          MMODD:
14220
14221                                          ;
14222                                          ;1/AC>FSRC=0
14223  071462  005037  002776                          CLR     @#FLAG          ;NO INTERRUPT
14224  071466  004767  001164                          JSR     R7,MDDSUB               ;DO TEST
14225  071472  012345  177777  177777                  .WORD   12345,-1,-1,-1  ;AC0
14226  071500  177777
14227  071502  000100  000000  000000                  .WORD   100,0,0,0       ;FSRC
14228  071510  000000
14229  071512  000000  000000  000000                  .WORD   0,0,0,0         ;FRACTIONAL RESULT
14230  071520  000000
14231  071522  000000  000000  000000                  .WORD   0,0,0,0         ;INTEGER RESULT
14232  071530  000000
14233  071532  000200                                  .WORD   200             ; TEST FPS
14234  071534  000204                                  .WORD   204             ;RESULTANT FPS
14235                                          ;2/AC=0<FSRC
14236  071536  005037  002776                          CLR     @#FLAG          ;NO INTERRUPT
14237  071542  004767  001110                          JSR     R7,MDDSUB               ;DO TEST
14238  071546  000000  000000  000000                  .WORD   0,0,0,0 ;AC0
14239  071554  000000
```

```
14240  071556  001234  177777  000000          .WORD   1234,-1,0,0     ;FSRC
14241  071564  000000
14242  071566  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL RESULT
14243  071574  000000
14244  071576  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
14245  071604  000000
14246  071606  007717                          .WORD   7717            ; TEST FPS
14247  071610  007704                          .WORD   7704            ;RESULTANT FPS
14248                          ;3/AC>FSRC>0
14249  071612  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14250  071616  004767  001034                  JSR     R7,MDDSUB               ;DO TEST
14251  071622  056252  125252  125252          .WORD   56252,125252,125252,125250      ;AC0
14252  071630  125250
14253  071632  040300  000000  000000          .WORD   40300,0,0,0     ;FSRC
14254  071640  000000
14255  071642  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL RESULT
14256  071650  000000
14257  071652  056377  177777  177777          .WORD   56377,-1,-1,-4          ;INTEGER RESULT
14258  071660  177774
14259  071662  000213                          .WORD   213             ; TEST FPS
14260  071664  000204                          .WORD   204             ;RESULTANT FPS
14261                          ;4/AC<0>FSRC
14262  071666  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14263  071672  004767  000760                  JSR     R7,MDDSUB               ;DO TEST
14264  071676  140240  000000  000000          .WORD   140240,0,0,0    ;AC0
14265  071704  000000
14266  071706  063714  146314  133572          .WORD   63714,146314,133572,167737      ;FSRC
14267  071714  167737
14268  071716  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL RESULT
14269  071724  000000
14270  071726  163777  177777  162531          .WORD   163777,-1,162531,125726         ;INTEGER RESULT
14271  071734  125726
14272  071736  000210                          .WORD   210             ; TEST FPS
14273  071740  000204                          .WORD   204             ;RESULTANT FPS
14274                          ;5/AC>FSRC>0
14275  071742  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14276  071746  004767  000704                  JSR     R7,MDDSUB               ;DO TEST
14277  071752  056200  000000  000000          .WORD   56200,0,0,1     ;AC0
14278  071760  000001
14279  071762  040340  000000  000000          .WORD   40340,0,0,0     ;FSRC
14280  071770  000000
14281  071772  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL RESULT
14282  072000  000000
14283  072002  056340  000000  000000          .WORD   56340,0,0,1             ;INTEGER RESULT
14284  072010  000001
14285  072012  000213                          .WORD   213             ; TEST FPS
14286  072014  000204                          .WORD   204             ;RESULTANT FPS
14287                          ;6/TRUNCATE
14288  072016  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14289  072022  004767  000630                  JSR     R7,MDDSUB               ;DO TEST
14290  072026  056252  125252  125252          .WORD   56252,125252,125252,125252              ;AC0
14291  072034  125252
14292  072036  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
14293  072044  000000
14294  072046  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14295  072054  000000
```

```
14296  072056  056377  177777  177777          .WORD   56377,-1,-1,-1          ;INTEGER RESULT
14297  072064  177777
14298  072066  000253                          .WORD   253             ; TEST FPS
14299  072070  000244                          .WORD   244             ;RESULT FPS
14300                          ;7/TRUNCATE FRACTION
14301  072072  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14302  072076  004767  000554                  JSR     R7,MODSUB               ;DO TEST
14303  072102  023252  125252  125252          .WORD   23252,125252,125252,125252              ;AC0
14304  072110  125252
14305  072112  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
14306  072120  000000
14307  072122  023377  177777  177777          .WORD   23377,-1,-1,-1          ;FRACTIONAL  RESULT
14308  072130  177777
14309  072132  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
14310  072140  000000
14311  072142  000253                          .WORD   253             ; TEST FPS
14312  072144  000240                          .WORD   240             ;RESULT FPS
14313                          ;8/ROUND INTEGER
14314  072146  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14315  072152  004767  000500                  JSR     R7,MODSUB               ;DO TEST
14316  072156  076600  000000  000000          .WORD   76600,0,0,125252                ;AC0
14317  072164  125252
14318  072166  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
14319  072174  000000
14320  072176  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14321  072204  000000
14322  072206  076700  000000  000000          .WORD   76700,0,0,-1            ;INTEGER RESULT
14323  072214  177777
14324  072216  000200                          .WORD   200             ; TEST FPS
14325  072220  000204                          .WORD   204             ;RESULT FPS
14326                          ;9/ROUND THROUGH FRACTION
14327  072222  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14328  072226  004767  000424                  JSR     R7,MODSUB               ;DO TEST
14329  072232  041525  052525  052525          .WORD   41525,052525,52525,52525                ;AC0
14330  072240  052525
14331  072242  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
14332  072250  000000
14333  072252  040177  177777  177777          .WORD   40177,-1,-1,177740              ;FRACTIONAL  RESULT
14334  072260  177740
14335  072262  041636  000000  000000          .WORD   41636,0,0,0             ;INTEGER RESULT
14336  072270  000000
14337  072272  007700                          .WORD   7700            ; TEST FPS
14338  072274  007700                          .WORD   7700            ;RESULT FPS
14339                          ;/OVERFLOW, TRAPS ENABLED
14340  072276  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
14341  072304  004767  000346                  JSR     R7,MODSUB               ;DO TEST
14342  072310  177777  177777  177777          .WORD   -1,-1,-1,-1             ;AC0
14343  072316  177777
14344  072320  040400  000000  000000          .WORD   40400,0,0,0             ;FSRC
14345  072326  000000
14346  072330  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14347  072336  000000
14348  072340  100177  177777  177777          .WORD   100177,-1,-1,-1         ;INTEGER RESULT
14349  072346  177777
14350  072350  007700                          .WORD   7700            ; TEST FPS
14351  072352  107706                          .WORD   107706          ;RESULT FPS
```

```
14352  072354  000010                          .WORD   10              ;FEC
14353                                  ;/INTEGER CHOPPED TO 56 BITS
14354  072356  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14355  072362  004767  000270                  JSR     R7,MODSUB               ;DO TEST
14356  072366  056700  000000  000000          .WORD   56700,0,0, 1            ;ACO
14357  072374  177777
14358  072376  044440  177777  177777          .WORD   44440,-1, 1, 1          ;FSRC
14359  072404  177777
14360  072406  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14361  072414  000000
14362  072416  063161  100000  000001          .WORD   63161,100000,1,40775            ;INTEGER RESULT
14363  072424  040775
14364  072426  000200                          .WORD   200             ; TEST FPS
14365  072430  000204                          .WORD   204             ;RESULT FPS
14366                                  ;/OVERFLOW, TRAPS DISABLED
14367  072432  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPT
14368  072440  004767  000212                  JSR     R7,MODSUB               ;DO TEST
14369  072444  066600  000000  000000          .WORD   66600,0,0,0             ;ACO
14370  072452  000000
14371  072454  066600  000000  000000          .WORD   66600,0,0,0             ;FSRC
14372  072462  000000
14373  072464  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14374  072472  000000
14375  072474  015200  000000  000000          .WORD   15200,0,0,0             ;INTEGER RESULT
14376  072502  000000
14377  072504  047700                          .WORD   47700           ; TEST FPS
14378  072506  147706                          .WORD   147706          ;RESULT FPS
14379  072510  000010                          .WORD   10              ;FEC
14380                                  ;/UNDERFLOW, TRAPS DISABLED
14381  072512  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPT
14382  072520  004767  000132                  JSR     R7,MODSUB               ;DO TEST
14383  072524  100277  000001  000002          .WORD   100277,1,2,-1           ;ACO
14384  072532  177777
14385  072534  100300  000001  000001          .WORD   100300,1,1,1            ;FSRC
14386  072542  000001
14387  072544  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
14388  072552  000000
14389  072554  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
14390  072562  000000
14391  072564  005200                          .WORD   5200            ; TEST FPS
14392  072566  005204                          .WORD   5204            ;RESULT FPS
14393  072570  000010                          .WORD   10              ;FEC
14394                                  ;/UNDERFLOW TRAPS ENABLED, UV AS RESULT
14395  072572  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
14396  072600  004767  000052                  JSR     R7,MODSUB               ;DO TEST
14397  072604  100277  000001  000002          .WORD   100277,1,2,-1           ;ACO
14398  072612  177777
14399  072614  100300  000001  000001          .WORD   100300,1,1,1            ;FSRC
14400  072622  000001
14401  072624  040417  040001  077403          .WORD   40417,40001,77403,0             ;FRACTIONAL  RESULT
14402  072632  000000
14403  072634  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
14404  072642  000000
14405  072644  002200                          .WORD   2200            ; TEST FPS
14406  072646  102200                          .WORD   102200          ;RESULT FPS
14407  072650  000012                          .WORD   12              ;FEC
```

```
14408                                   ;
14409
14410  072652  000167  000244                   JMP     MOP15           ;JUMP OVER SUBROUTINE
14411                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14412                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14413                                   ;
14414                                   ;               AC0
14415                                   ;               FSRC
14416                                   ;               FRACTIONAL RESULT
14417                                   ;               INTEGER RESULT
14418                                   ;               FPS BEFORE EXECUTION
14419                                   ;               FPS AFTER EXECUTION
14420                                   ;               (FEC)
14421                                   ;
14422                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14423                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14424                                   ;
14425  072656  012605                   MODSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14426  072660  012737  072742  000244           MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14427  072666  012702  000200                   MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14428  072672  170102                           LDFPS   R2                      ;LOAD FPS
14429  072674  172415                           LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14430  072676  012701  071452                   MOV     #MODGAR,R1              ;LOAD KNOWN INTO AC1
14431  072702  172511                           LDD     (R1),AC1                ;
14432  072704  010501                           MOV     R5,R1                   ;POINT TO FSRC DATA
14433  072706  062701  000010                   ADD     #10,R1
14434  072712  016502  000040                   MOV     40(R5),R2               ;GET TEST FPS
14435  072716  170102                           LDFPS   R2                      ;LOAD TEST FPS
14436                                   ;
14437  072720  171411                           MODD    (R1),AC0                ;*TEST INSTRUCTION
14438  072722  170011                   1$:     SETD                            ;WAIT FOR POSSIBLE FPA TRAP.
14439                                   ;
14440                                   ;INSTRUCTION DIDNT TRAP
14441  072724  032737  000001  002776           BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
14442  072732  001420                           BEQ     2$                      ;BRANCH IF GOOD
14443  072734  104003                           ERROR   .3              ;FPP ERROR
14444                                                                   ;INSTRUCTION SHOULD HAVE TRAPPED
14445  072736  000167  000032                   JMP     2$                      ;REJOIN CODE
14446                                   ;
14447                                   ;INSTRUCTION TRAPPED
14448  072742  032737  000001  002776   50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14449  072750  001003                           BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14450  072752  104003                           ERROR   .3              ;FPP ERROR
14451                                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
14452  072754  000167  000014                   JMP     2$                      ;REJOIN CODE
14453  072760  012604                   51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14454  072762  005726                           TST     (SP)+                   ;CLEAN UP STACK
14455  072764  022704  072722                   CMP     #1$,R4                  ;
14456  072770  001401                           BEQ     2$                      ;BRANCH IF GOOD COMPARE
14457  072772  104003                           ERROR   .3              ;FPP ERROR
14458                                                                   ;PC WAS INCORRECT
14459                                   ;
14460                                   ;COMMON CODE FOR TRAP AND NO TRAP
14461  072774  170203                   2$:     STFPS   R3                      ;SAVE FPS
14462  072776  012702  000200                   MOV     #200,R2                 ;SET FPP TO DOUBLE
14463  073002  170102                           LDFPS   R2
```

```
14464  073004  012701  003102          MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14465                                ;SAVE FRACTIONAL RESULT
14466  073010  174011                  STD     AC0,(R1)                ;SAVE AC0  RESULT
14467  073012  026503  000042          CMP     42(R5),R3               ;VERIFY STATUS
14468  073016  001401                  BEQ     3$                      ;BRANCH IF GOOD
14469  073020  104003                  ERROR   .3                  ;FPP ERROR
14470                                                               ;BAD FPS
14471  073022  010504          3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14472  073024  062704  000020          ADD     #20,R4
14473  073030  004767  037636  4$:     JSR     R7,DATVER                       ;VERIFY DATA
14474  073034  005767  110020          TST     COUNT
14475  073040  001401                  BEQ     5$                      ;BRANCH IF GOOD
14476  073042  104003                  ERROR   .3                  ;FPP ERROR
14477                                                               ;BAD AC0
14478                                ;SAVE INTEGER RESULT
14479  073044  174111          5$:     STD     AC1,(R1)                ;SAVE AC1 RESULT
14480  073046  010504                  MOV     R5,R4               ;POINT TO EXPECTED
14481  073050  062704  000030          ADD     #30,R4
14482  073054  004767  037612          JSR     R7,DATVER               ;VERIFY DATA
14483  073060  005767  107774          TST     COUNT                   ;
14484  073064  001401                  BEQ     6$                      ;BRANCH IF GOOD
14485  073066  104003                  ERROR   .3                  ;FPP ERROR
14486                                                               ;BAD AC1
14487  073070  005737  002776  6$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14488  073074  001002                  BNE     7$                      ;BRANCH IF NEED TO CHECK
14489  073076  000165  000044          JMP     44(R5)                  ;RETURN FROM TEST
14490  073102  170301          7$:     STST    R1                      ;SAVE FEC
14491  073104  016504  000044          MOV     44(R5),R4               ;GET FEC
14492  073110  020401                  CMP     R4,R1                   ;VERIFY FEC
14493  073112  001401                  BEQ     8$                      ;BRANCH IF GOOD
14494  073114  104003                  ERROR   .3                  ;FPP ERROR
14495                                                                       ;BAD FEC
14496  073116  000165  000046  8$:     JMP     46(R5)              ;RETURN FROM TEST
14497                                ;
14498  073122                       MOP15:
14499                                ;
14500                                ;
14501                                ;---------------------------------
14502                                ;TEST STCFD
14503                                ;
14504                                        ;
14505                                        ;
14506                                        ;
14507  073122                       MSFD:
14508
14509                                ;
14510                                ;1/AC=0
14511  073122  004767  000170                  JSR     R7,SFDSUB                       ;DO TEST
14512  073126  000177  000000  000000          .WORD   0177,0,0,1                      ;AC0
14513  073134  000001
14514  073136  000000  000000  000000          .WORD   0,0,0,0             ;RESULT
14515  073144  000000
14516  073146  047557                          .WORD   47557               ; TEST FPS
14517  073150  047544                          .WORD   47544               ;RESULT FPS
14518                                ;2/AC>0, TRUNCATE
14519  073152  004767  000140                  JSR     R7,SFDSUB           ;DO TEST
```

```
14520  073156  077577  177777  177777          .WORD   77577, 1,-1, 1          ;AC0
14521  073164  177777
14522  073166  077577  177777  000000          .WORD   77577,-1,0,0             ;RESULT
14523  073174  000000
14524  073176  007540                          .WORD   7540             ; TEST FPS
14525  073200  007540                          .WORD   7540             ;RESULT FPS
14526                          ;3/AC<0, ROUND
14527  073202  004767  000110                  JSR     R7,SFDSUB                ;DO TEST
14528  073206  100377  177777  100000          .WORD   100377,-1,100000,0               ;AC0
14529  073214  000000
14530  073216  100377  177777  000000          .WORD   100377,-1,0,0    ;RESULT
14531  073224  000000
14532  073226  007517                          .WORD   7517             ; TEST FPS
14533  073230  007510                          .WORD   7510             ;RESULT FPS
14534                          ;4/AC=-0
14535  073232  004767  000060                  JSR     R7,SFDSUB                ;DO TEST
14536  073236  100000  000000  000000          .WORD   100000,0,0,0             ;AC0
14537  073244  000000
14538  073246  000000  000000  000000          .WORD   0,0,0,0          ;RESULT
14539  073254  000000
14540  073256  007757                          .WORD   7757             ; TEST FPS
14541  073260  007744                          .WORD   7744             ;RESULT FPS
14542                          ;5/AC<0
14543  073262  004767  000030                  JSR     R7,SFDSUB                ;DO TEST
14544  073266  125252  125252  125252          .WORD   125252,125252,125252,125252              ;AC0
14545  073274  125252
14546  073276  125252  125252  000000          .WORD   125252,125252,0,0                ;RESULT
14547  073304  000000
14548  073306  000000                          .WORD   0                ; TEST FPS
14549  073310  000010                          .WORD   10               ;RESULT FPS
14550                          ;
14551
14552  073312  000167  000104                  JMP     @OP16                    ;GET OVER SUBROUTINE
14553                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14554                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14555                          ;STCFD
14556                          ;               AC0
14557                          ;               RESULT
14558                          ;               FPS BEFORE EXECUTION
14559                          ;               FPS AFTER EXECUTION
14560                          ;
14561                          ;THERE CAN BE NO TRAPS WITH THE STCFD INSTRUCTION
14562                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14563                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14564                          ;
14565  073316  012605                  SFDSUB: MOV     (SP)+,R5                 ; RETURN ADDRESS TO USE AS POINTER
14566  073320  012737  073414  000244          MOV     @#501,@#FPVEC            ;REDIRECT TRAP VECTOR
14567  073326  012702  000200                  MOV     #200,R2                  ;SET TO DOUBLE MODE FOR LOAD
14568  073332  170102                          LDFPS   R2                       ;LOAD FPS
14569  073334  172415                          LDD     (R5),AC0                 ;LOAD AC0 WITH TEST DATA
14570  073336  012701  003102                  MOV     #RECDST,R1               ;POINT TO RESULT AREA
14571  073342  016502  000020                  MOV     20(R5),R2                ;GET TEST FPS
14572  073346  170102                          LDFPS   R2                       ;LOAD TEST FPS
14573                          ;
14574  073350  176011                  401:    STCFD   AC0,(R1)                 ;*TEST INSTRUCTION
14575                          ;
```

```
14576                                  ;INSTRUCTION DIDNT TRAP
14577                                  ;VERIFY STATUS
14578  073352  170203                  2$:     STFPS   R3                      ;SAVE FPS
14579  073354  016502  000022                  MOV     22(R5),R2                       ;GET EXPECTED STATUS
14580  073360  020203                          CMP     R2,R3                   ;VERIFY STATUS
14581  073362  001401                          BEQ     3$                      ;BRANCH IF GOOD
14582  073364  104003                          ERROR   .3              ;FPP ERROR
14583                                                                  ;BAD FPS
14584  073366  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14585  073370  062704  000010                  ADD     #10,R4
14586  073374  004767  037272          4$:     JSR     R7,DATVER                       ;VERIFY DATA
14587  073400  005767  107454                  TST     COUNT
14588  073404  001401                          BEQ     5$                      ;BRANCH IF GOOD
14589  073406  104003                          ERROR   .3              ;FPP ERROR
14590                                                                  ;BAD ACO
14591  073410  000165  000024          5$:     JMP     24(R5)                  ;RETURN FROM TEST
14592                                  ;INSTRUCTION TRAPPED
14593  073414  104003                  50$:    ERROR   .3              ;FPP ERROR
14594                                                                  ;INSTRUCTION WASNT SUPPOSE TO TRAP
14595  073416  000165  000024                  JMP     24(R5)          ;RETURN FROM TEST
14596                                  ;
14597  073422                          MOP16:
14598
14599                                  ;
14600                                  ;
14601                                  ;---------------------------------
14602                                  ;TEST STCDF
14603                                  ;
14604                                          ;
14605                                          ;
14606                                          ;
14607  073422                          MSDF:
14608
14609                                  ;
14610                                  ;1/AC=0
14611  073422  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14612  073426  004767  000220                  JSR     R7,SDFSUB               ;DO TEST
14613  073432  000177  000000  000000          .WORD   177,0,0,0               ;ACO
14614  073440  000000
14615  073442  000000  000000                  .WORD   0,0             ;RESULT
14616  073446  000200                          .WORD   200             ; TEST FPS
14617  073450  000204                          .WORD   204             ;RESULT FPS
14618                                  ;2/AC=-0
14619  073452  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14620  073456  004767  000170                  JSR     R7,SDFSUB               ;DO TEST
14621  073462  100000  000300  000200          .WORD   100000,300,200,100              ;ACO
14622  073470  000100
14623  073472  000000  000000                  .WORD   0,0             ;RESULT
14624  073476  007777                          .WORD   7777            ; TEST FPS
14625  073500  007744                          .WORD   7744            ;RESULT FPS
14626                                  ;3/AC>0, TRUNCATE
14627  073502  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
14628  073506  004767  000140                  JSR     R7,SDFSUB               ;DO TEST
14629  073512  055555  055555  177777          .WORD   55555,55555,-1,-1               ;ACO
14630  073520  177777
14631  073522  055555  055555                  .WORD   55555,55555             ;RESULT
```

```
073526  000240                                  .WORD   240                     ; TEST FPS
073530  00024u                                  .WORD   240                     ;RESULT FPS
                                        ;4/AC<0, ROUND TO UNDEFINED VARIABLE
073532  012737  000001  002776                  MOV     #1,@#FLAG                       ;INTERRUPT
073540  004767  000106                          JSR     R7,SDFSUB                       ;DO TEST
073544  077777  177777  100000                  .WORD   77777,-1,100000,0                       ;ACO
073552  000000
073554  000000  000000                          .WORD   0,0                     ;RESULT
073560  001200                                  .WORD   1200                    ; TEST FPS
073562  101206                                  .WORD   101206                  ;RESULT FPS
                                        ;5/AC<0, ROUND
073564  005037  002776                          CLR     @#FLAG                  ;NO INTERRUPT
073570  004767  000056                          JSR     R7,SDFSUB                       ;DO TEST
073574  125252  125252  125252                  .WORD   125252,125252,125252,125252                     ;ACO
073602  125252
073604  125252  125253                          .WORD   125252,125253                   ;RESULT
073610  007700                                  .WORD   7700                    ; TEST FPS
073612  007710                                  .WORD   7710                    ;RESULT FPS
                                        ;6/ROUND TO UV, TRAPS DISABLED
073614  012737  000002  002776                  MOV     #2,@#FLAG                       ;INTERRUPT
073622  004767  000024                          JSR     R7,SDFSUB                       ;DO TEST
073626  077777  177777  177777                  .WORD   77777,-1,-1,0                   ;ACO
073634  000000
073636  000000  000000                          .WORD   0,0                     ;RESULT
073642  006700                                  .WORD   6700                    ; TEST FPS
073644  006706                                  .WORD   6706                    ;RESULT FPS
                                        ;

073646  000167  000202                          JMP     HOP17                   ;GET OVER SUBROUTINE
                                        ;
                                        ;
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;STCDF
                                        ;                       ACO
                                        ;                       RESULT
                                        ;                       FPS BEFORE EXECUTION
                                        ;                       FPS AFTER EXECUTION
                                        ;
                                        ;A TRAP CAN ONLY OCCUR IF ROUNDING CAUSES OVERFLOW
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;
073652  012605                          SDFSUB: MOV     (SP)+,R5                        ; RETURN ADDRESS TO USE AS POINTER
073654  012737  073730  000244                  MOV     #50$,@#FPVEC                    ;REDIRECT TRAP VECTOR
073662  012702  000200                          MOV     #200,R2                         ;SET TO DOUBLE MODE FOR LOAD
073666  170102                                  LDFPS   R2                              ;LOAD FPS
073670  172415                                  LDD     (R5),AC0                        ;LOAD AC0 WITH TEST DATA
073672  012701  003102                          MOV     #RECDST,R1                      ;POINT TO RESULT AREA
073676  016502  000014                          MOV     14(R5),R2                       ;GET TEST FPS
073702  170102                                  LDFPS   R2                              ;LOAD TEST FPS
                                        ;
073704  176011                          40$:    STCDF   AC0,(R1)                        ;*TEST INSTRUCTION
073706  170327                          1$:     STST    (PC)+                           ;WAIT FOR POSSIBLE FPA TRAP.
073710  000000                                  .WORD   0                               ;STORE STATUS HERE.
                                        ;
```

```
14688                                  ;INSTRUCTION DIDNT TRAP
14689  073712  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
14690  073720  001420                          BEQ     2$                      ;BRANCH IF GOOD
14691  073722  104003                          ERROR   .+3             ;FPP ERROR
14692                                                                  ;INSTRUCTION SHOULD HAVE TRAPPED
14693  073724  000167  000032                  JMP     2$                      ;REJOIN CODE
14694                                  ;
14695                                  ;INSTRUCTION TRAPPED
14696  073730  032737  000001  002776  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14697  073736  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14698  073740  104003                          ERROR   .+3             ;FPP ERROR
14699                                                                  ;INSTRUCTION WASNT SUPPOSE TO TRAP
14700  073742  000167  000014                  JMP     2$                      ;REJOIN CODE
14701  073746  012604                  51$:     MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14702  073750  005726                          TST     (SP)+                   ;CLEAN UP STACK
14703  073752  022704  073706                  CMP     #1$,R4                  ;
14704  073756  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
14705  073760  104003                          ERROR   .+3             ;FPP ERROR
14706                                                                  ;PC WAS INCORRECT
14707                                  ;
14708                                  ;COMMON CODE FOR TRAP AND NO TRAP
14709                                  ;VERIFY STATUS
14710  073762  170203                  2$:     STFPS   R3                      ;SAVE FPS
14711  073764  016502  000016                  MOV     16(R5),R2                       ;GET EXPECTED STATUS
14712  073770  020203                          CMP     R2,R3                   ;VERIFY STATUS
14713  073772  001401                          BEQ     3$                      ;BRANCH IF GOOD
14714  073774  104003                          ERROR   .+3             ;FPP ERROR
14715                                                                  ;BAD FPS
14716  073776  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14717  074000  062704  000010                  ADD     #10,R4
14718  074004  004767  036644          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
14719  074010  005767  107044                  TST     COUNT
14720  074014  001401                          BEQ     5$                      ;BRANCH IF GOOD
14721  074016  104003                          ERROR   .+3             ;FPP ERROR
14722                                                                  ;BAD ACO
14723  074020  005737  002776          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14724  074024  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
14725  074026  000165  000020                  JMP     20(R5)                  ;RETURN FROM TEST
14726                                  ;VERIFY FEC
14727  074032  012704  003062          7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
14728  074036  170314                          STST    (R4)                    ;SAVE FEC
14729  074040  021427  000010                  CMP     (R4),#10                        ;VERIFY FEC FOR OVERFLOW
14730  074044  001401                          BEQ     8$                      ;BRANCH IF GOOD
14731  074046  104003                          ERROR   .+3             ;FPP ERROR
14732                                                                  ;BAD FEC
14733  074050  000165  000020          8$:     JMP     20(R5)          ;RETURN FROM TEST
14734                                  ;
14735  074054                          MOP17:
14736                                  ;
14737                                  ;
14738                                  ;---------------------------------
14739                                  ;TEST STCFD - USING ILLEGAL ACCUMULATOR
14740                                  ;
14741                                          ;
14742                                          ;
14743                                          ;
```

```
14744  074054                          MSFDI:
14745
14746                                  ;
14747  074054  012701  040000                  MOV     #40000,R1               ;DISABLE INTERRUPTS
14748  074060  170101                          LDFPS   R1                      ;
14749  074062  176006                          STCFD   AC0,AC6                 ;*TEST ILLEGAL INSTRUCTION
14750  074064  170202                          STFPS   R2                      ;SAVE STATUS
14751  074066  170303                          STST    R3                      ;SAVE FEC
14752  074070  022702  140000                  CMP     #140000,R2              ;VERIFY FER SET
14753  074074  001401                          BEQ     1$                      ;BRANCH IF ERROR RECEIVED
14754  074076  104003                          ERROR   .+3                 ;FPP ERROR
14755                                                                      ;FER BIT NOT SET ON ILLEGAL INST.
14756  074100  022703  000002          1$:     CMP     #2,R3                   ;VERIFY FEC = FLOATING OPCDOE ERROR
14757  074104  001401                          BEQ     2$                      ;BRANCH IF GOOD
14758  074106  104003                          ERROR   .+3                 ;FPP ERROR
14759                                                                      ;FEC INCORRECT
14760  074110                          2$:
14761
14762                                  ;
14763                                  ;
14764                                  ;
14765                                  ;---------------------------------
14766                                  ;TEST CLRD
14767                                  ;
14768                                          ;
14769                                          ;
14770                                          ;
14771  074110                          MCLRD:
14772
14773                                  ;
14774  074110  012701  003724                  MOV     #TAB47,R1               ;POINT TO DATA
14775  074114  012704  000200                  MOV     #200,R4                 ;SET FPP STATUS TO DOUBLE
14776  074120  170104                          LDFPS   R4
14777  074122  172411                          LDD     (R1),AC0                ;
14778  074124  012701  003102                  MOV     #RECDST,R1              ;POINT TO DATA BUFFER
14779  074130  174011                          STD     AC0,(R1)                ;STORE GARBAGE
14780  074132  170411                          CLRD    (R1)                    ;CLEAR DATA BUFFER
14781  074134  012704  003254                  MOV     #TAB6,R4                ;VERIFY BUFFER =0
14782  074140  004767  036526                  JSR     R7,DATVER               ;
14783  074144  005767  106710                  TST     COUNT                   ;
14784  074150  001401                          BEQ     1$                      ;BRANCH I RECDST = 0
14785  074152  104003                          ERROR   .+3                 ;FPP ERROR
14786                                                                      ;RECDST NOT CLEARED
14787  074154  170202          1$:     STFPS   R2                              ;SAVE STATUS
14788  074156  020227  000204                  CMP     R2,#204                 ;VERIFY STATUS
14789  074162  001401                          BEQ     2$                      ;BRANCH IF GOOD
14790  074164  104003                          ERROR   .+3                 ;FPP ERROR
14791                                                                      ;BAD STATUS
14792  074166                          2$:
14793
14794
14795                                  ;
14796                                  ;
14797                                  ;---------------------------------
14798                                  ;TEST CLRD, ILLEGAL ACCUMULATOR
14799                                  ;
```

```
14800                                           ;
14801                                           ;
14802                                           ;
14803  074166                           MCLRI:
14804
14805                                           ;
14806  074166  012704  040200                  MOV     #40200,R4               ;DISABLE INTERRUPTS
14807  074172  170104                          LDFPS   R4                      ;LOAD STATUS
14808  074174  170406                          CLRD    R6                      ;*TEST INSTRUCTION WITH ILLEGAL ACC
14809  074176  170203                          STFPS   R3                      ;SAVE STATUS
14810  074200  170305                          STST    R5                      ;SAVE FEC
14811  074202  022703  140200                  CMP     #140200,R3              ;VERIFY ERROR
14812  074206  001401                          BEQ     1$                      ;BRANCH IF FER SET
14813  074210  104003                          ERROR   +3              ;FPP ERROR
14814                                                                   ;ERROR IN FPS
14815  074212  022705  000002          1$:     CMP     #2,R5           ;VERIFY FEC =2 OPCODE ERROR
14816  074216  001401                          BEQ     2$              ;BRANCH IF GOOD
14817  074220  104003                          ERROR   +3              ;FPP ERROR
14818                                                                   ;BAD FEC
14819  074222                           2$:
14820
14821
14822                                           ;
14823                                           ;
14824                                           ;---------------------------------
14825                                           ;TEST LDFPS, STFPS MODE 1
14826                                           ;
14827                                                   ;
14828                                                   ;
14829                                                   ;
14830  074222                           MLS1:
14831
14832                                           ;
14833  074222  012704  003122                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
14834  074226  012714  147757                  MOV     #147757,(R4)            ;SETUP EXPECTED STATUS
14835  074232  012701  003072                  MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
14836  074236  012737  074306  000244          MOV     #10$,@#PVEC             ;SETUP TRAP VECTOR
14837  074244  170114                          LDFPS   (R4)                    ;*TEST INSTRUCTION
14838  074246  170211                          STFPS   (R1)                    ;*TEST INSTRUCTION
14839  074250  020427  003122                  CMP     R4,#TSTLOC              ;VERIFY R4
14840  074254  001401                          BEQ     1$                      ;BRANCH IF GOOD
14841  074256  104003                          ERROR   +3              ;FPP ERROR
14842                                           ;
14843  074260  020127  003072          1$:     CMP     R1,#RECST               ;VERIFY R1
14844  074264  001401                          BEQ     2$                      ;BRANCH IF GOOD
14845  074266  104003                          ERROR   +3              ;FPP ERROR
14846                                                                   ;BAD R1
14847  074270  023727  003072  147757  2$:     CMP     @#RECST,#147757         ;VERIFY STATUS
14848  074276  001406                          BEQ     3$                      ;BRANCH F GOOD
14849  074300  104003                          ERROR   +3              ;FPP ERROR
14850                                                                   ;BAD STATUS\
14851  074302  000167  000006                  JMP     3$                      ;GET OVER TRAP
14852                                   ;UNEXPECTED TRAP
14853  074306  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
14854  074310  012605                          MOV     (SP)+,R5                        ;SAVE PS
14855  074312  104003                          ERROR   +3              ;FPP ERROR
```

```
14856
14857  074314                          3$:                                     ;UNEXPECTED TRAP
14858
14859
14860                                  ;
14861                                  ;
14862                                  ;---------------------------------
14863                                  ;TEST LDFPS, STFPS MODE 2
14864                                  ;
14865                                          ;
14866                                          ;
14867                                          ;
14868  074314                          MLS2:
14869
14870                                  ;
14871  074314  012704  003122                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
14872  074320  012714  145557                  MOV     #145557,(R4)            ;SETUP EXPECTED STATUS
14873  074324  012701  003072                  MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
14874  074330  012737  074400  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
14875  074336  170124                          LDFPS   (R4)+                   ;*TEST INSTRUCTION
14876  074340  170221                          STFPS   (R1)+                   ;*TEST INSTRUCTION
14877  074342  020427  003124                  CMP     R4,#TSTLOC+2            ;VERIFY R4
14878  074346  001401                          BEQ     1$                      ;BRANCH IF GOOD
14879  074350  104003                          ERROR   +3                      ;FPP ERROR
14880                                                                          ;
14881  074352  020127  003074          1$:     CMP     R1,#RECST+2                     ;VERIFY R1
14882  074356  001401                          BEQ     2$                              ;BRANCH IF GOOD
14883  074360  104003                          ERROR   +3                      ;FPP ERROR
14884                                                                          ;BAD R1
14885  074362  023727  003072  145557  2$:     CMP     @#RECST,#145557                 ;VERIFY STATUS
14886  074370  001406                          BEQ     3$                              ;BRANCH F GOOD
14887  074372  104003                          ERROR   +3                      ;FPP ERROR
14888                                                                          ;BAD STATUS\
14889  074374  000167  000006                  JMP     3$                              ;GET OVER TRAP
14890                                  ;UNEXPECTED TRAP
14891  074400  012600                  10$:    MOV     (SP)+,R0                                ;SAVE PC
14892  074402  012605                          MOV     (SP)+,R5                                ;SAVE PS
14893  074404  104003                          ERROR   +3                      ;FPP ERROR
14894                                                                          ;UNEXPECTED TRAP
14895  074406                          3$:
14896
14897
14898                                  ;
14899                                  ;
14900                                  ;---------------------------------
14901                                  ;TEST LDFPS, STFPS MODE 3
14902                                  ;
14903                                          ;
14904                                          ;
14905                                          ;
14906  074406                          MLS3:
14907
14908                                  ;
14909  074406  012704  003122                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
14910  074412  012737  003126  003122          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
14911  074420  012737  147501  003126          MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
```

```
14912  074426  012701  003132                  MOV     #TSTLOC+10,R1           ;R1 POINTS TO TSTLOC+10
14913  074432  012737  003072  003132          MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
14914  074440  012737  074510  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
14915  074446  170134                          LDFPS   @(R4)+                  ;*TEST INSTRUCTION
14916  074450  170231                          STFPS   @(R1)+                  ;*TEST INSTRUCTION
14917  074452  020427  003124                  CMP     R4,#TSTLOC+2            ;VERIFY R4
14918  074456  001401                          BEQ     1$                      ;BRANCH IF GOOD
14919  074460  104003                          ERROR   +3              ;FPP ERROR
14920                                                                  ;
14921  074462  020127  003134          1$:     CMP     R1,#TSTLOC+12           ;VERIFY R1
14922  074466  001401                          BEQ     2$                      ;BRANCH IF GOOD
14923  074470  104003                          ERROR   +3              ;FPP ERROR
14924                                                                  ;BAD R1
14925  074472  023727  003072  147501  2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
14926  074500  001406                          BEQ     3$                      ;BRANCH F GOOD
14927  074502  104003                          ERROR   +3              ;FPP ERROR
14928                                                                  ;BAD STATUS\
14929  074504  000167  000006                  JMP     3$                      ;GET OVER TRAP
14930                                  ;UNEXPECTED TRAP
14931  074510  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
14932  074512  012605                          MOV     (SP)+,R5                        ;SAVE PS
14933  074514  104003                          ERROR   +3              ;FPP ERROR
14934                                                                  ;UNEXPECTED TRAP
14935  074516                          3$:
14936
14937
14938                                  ;
14939                                  ;
14940                                  ;---------------------------------
14941                                  ;TEST LDFPS, STFPS MODE 4
14942                                  ;
14943                                          ;
14944                                          ;
14945                                          ;
14946  074516                          MLS4:
14947
14948                                  ;
14949  074516  012704  003124                  MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
14950  074522  012737  147757  003122          MOV     #147757,@#TSTLOC        ;TSTLOC= STATUS ADDRESS
14951  074530  012701  003074                  MOV     #RECST+2,R1             ;SET BUFFER FOR RECEIVED STATUS
14952  074534  012737  074604  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
14953  074542  170144                          LDFPS   -(R4)                   ;*TEST INSTRUCTION
14954  074544  170241                          STFPS   -(R1)                   ;*TEST INSTRUCTION
14955  074546  020427  003122                  CMP     R4,#TSTLOC              ;VERIFY R4
14956  074552  001401                          BEQ     1$                      ;BRANCH IF GOOD
14957  074554  104003                          ERROR   +3              ;FPP ERROR
14958                                                                  ;
14959  074556  020127  003072          1$:     CMP     R1,#RECST               ;VERIFY R1
14960  074562  001401                          BEQ     2$                      ;BRANCH IF GOOD
14961  074564  104003                          ERROR   +3              ;FPP ERROR
14962                                                                  ;BAD R1
14963  074566  023727  003072  147757  2$:     CMP     @#RECST,#147757         ;VERIFY STATUS
14964  074574  001406                          BEQ     3$                      ;BRANCH F GOOD
14965  074576  104003                          ERROR   +3              ;FPP ERROR
14966                                                                  ;BAD STATUS\
14967  074600  000167  000006                  JMP     3$                      ;GET OVER TRAP
```

```
14968                                  ;UNEXPECTED TRAP
14969  074604  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
14970  074606  012605                          MOV     (SP)+,R5                        ;SAVE PS
14971  074610  104003                          ERROR   +3                      ;FPP ERROR
14972                                                                          ;UNEXPECTED TRAP
14973  074612                          3$:
14974
14975
14976                                  ;
14977                                  ;
14978                                  ;---------------------------------
14979                                  ;TEST LDFPS, STFPS MODE 5
14980                                  ;
14981                                          ;
14982                                          ;
14983                                          ;
14984  074612                          MLS5:
14985
14986                                  ;
14987  074612  012704  003124                  MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
14988  074616  012737  003126  003122          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
14989  074624  012737  147501  003126          MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
14990  074632  012701  003134                  MOV     #TSTLOC+12,R1           ;R1 POINTS TO 412
14991  074636  012737  003072  003132          MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
14992  074644  012737  074714  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
14993  074652  170154                          LDFPS   @-(R4)                  ;*TEST INSTRUCTION
14994  074654  170251                          STFPS   @-(R1)                  ;*TEST INSTRUCTION
14995  074656  020427  003122                  CMP     R4,#TSTLOC              ;VERIFY R4
14996  074662  001401                          BEQ     1$                      ;BRANCH IF GOOD
14997  074664  104003                          ERROR   +3              ;FPP ERROR
14998                                                                  ;
14999  074666  020127  003132          1$:     CMP     R1,#TSTLOC+10                   ;VERIFY R1
15000  074672  001401                          BEQ     2$                      ;BRANCH IF GOOD
15001  074674  104003                          ERROR   +3              ;FPP ERROR
15002                                                                  ;BAD R1
15003  074676  023727  003072  147501  2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
15004  074704  001406                          BEQ     3$                      ;BRANCH F GOOD
15005  074706  104003                          ERROR   +3              ;FPP ERROR
15006                                                                  ;BAD STATUS\
15007  074710  000167  000006                  JMP     3$                      ;GET OVER TRAP
15008                                  ;UNEXPECTED TRAP
15009  074714  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15010  074716  012605                          MOV     (SP)+,R5                        ;SAVE PS
15011  074720  104003                          ERROR   +3              ;FPP ERROR
15012                                                                  ;UNEXPECTED TRAP
15013  074722                          3$:
15014
15015
15016                                  ;
15017                                  ;
15018                                  ;---------------------------------
15019                                  ;TEST LDFPS, STFPS MODE 6
15020                                  ;
15021                                          ;
15022                                          ;
15023                                          ;
```

```
 15024  074722                          MLS6:
 15025
 15026                                  ;
 15027  074722  012704  003122                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
 15028  074726  012737  140001  003126          MOV     #140001,@#TSTLOC+4      ;SETUP EXPECTED STATUS
 15029  074734  012701  003232                  MOV     #TSTLOC+110,R1          ;R1 WILL POINT TO TESTLOC+10
 15030  074740  012737  075014  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
 15031  074746  170164  000004                  LDFPS   4(R4)                   ;*TEST INSTRUCTION
 15032  074752  170261  177700                  STFPS   -100(R1)                        ;*TEST INSTRUCTION
 15033  074756  020427  003122                  CMP     R4,#TSTLOC              ;VERIFY R4
 15034  074762  001401                          BEQ     1$                      ;BRANCH IF GOOD
 15035  074764  104003                          ERROR   +3              ;FPP ERROR
 15036                                                                  ;
 15037  074766  020127  003232          1$:     CMP     R1,#TSTLOC+110          ;VERIFY R1
 15038  074772  001401                          BEQ     2$                      ;BRANCH IF GOOD
 15039  074774  104003                          ERROR   +3              ;FPP ERROR
 15040                                                                  ;BAD R1
 15041  074776  023727  003132  140001  2$:     CMP     @#TSTLOC+10,#140001     ;VERIFY STATUS
 15042  075004  001406                          BEQ     3$                      ;BRANCH F GOOD
 15043  075006  104003                          ERROR   +3              ;FPP ERROR
 15044                                                                  ;BAD STATUS\
 15045  075010  000167  000006                  JMP     3$                      ;GET OVER TRAP
 15046                                  ;UNEXPECTED TRAP
 15047  075014  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
 15048  075016  012605                          MOV     (SP)+,R5                        ;SAVE PS
 15049  075020  104003                          ERROR   +3              ;FPP ERROR
 15050                                                                  ;UNEXPECTED TRAP
 15051  075022                          3$:
 15052
 15053
 15054                                  ;
 15055                                  ;
 15056                                  ;---------------------------------
 15057                                  ;TEST LDFPS, STFPS MODE 7
 15058                                  ;
 15059                                          ;
 15060                                          ;
 15061                                          ;
 15062  075022                          MLS7:
 15063
 15064                                  ;
 15065  075022  012704  003222                  MOV     #TSTLOC+100,R4          ;POINT R4 TO RAM
 15066  075026  012737  003126  003122          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
 15067  075034  012737  145501  003126          MOV     #145501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
 15068  075042  012701  003032                  MOV     #TSTLOC-70,R1           ;R1 POINTS TO TSTLOC+10
 15069  075046  012737  003132  003124          MOV     #TSTLOC+10,@#TSTLOC+2   ;
 15070  075054  012737  075130  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
 15071  075062  170174  177700                  LDFPS   @-100(R4)                       ;*TEST INSTRUCTION
 15072  075066  170271  000072                  STFPS   @72(R1)                 ;*TEST INSTRUCTION
 15073  075072  020427  003222                  CMP     R4,#TSTLOC+100          ;VERIFY R4
 15074  075076  001401                          BEQ     1$                      ;BRANCH IF GOOD
 15075  075100  104003                          ERROR   +3              ;FPP ERROR
 15076                                                                  ;
 15077  075102  020127  003032          1$:     CMP     R1,#TSTLOC-70           ;VERIFY R1
 15078  075106  001401                          BEQ     2$                      ;BRANCH IF GOOD
 15079  075110  104003                          ERROR   +3              ;FPP ERROR
```

```
15080
15081  075112  023727  003132  145501  2$:     CMP     @#TSTLOC+10,#145501     ;BAD R1
15082  075120  001406                          BEQ     3$                      ;VERIFY STATUS
15083  075122  104003                          ERROR   .3                      ;BRANCH F GOOD
15084                                                                         ;FPP ERROR
15085  075124  000167  000006                  JMP     3$                      ;BAD STATUS\
15086                                                                                 ;GET OVER TRAP
                                ;UNEXPECTED TRAP
15087  075130  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15088  075132  012605                          MOV     (SP)+,R5                        ;SAVE PS
15089  075134  104003                          ERROR   .3                      ;FPP ERROR
15090                                                                         ;UNEXPECTED TRAP
15091  075136                          3$:
15092
15093
15094                                ;
15095                                ;
15096                                ;--------------------------------
15097                                ;TEST LDCLD MODE 27
15098                                ;
15099                                        ;
15100                                        ;
15101                                        ;
15102  075136                        MLDC2:
15103
15104                                ;
15105  075136  005001                        CLR     R1                      ;INIT R1
15106  075140  012704  007700                MOV     #7700,R4                ;FPS=DOUBLE, LONG
15107  075144  170104                        LDFPS   R4                      ;
15108  075146  012737  075202  000244        MOV     #10$,@#FPVEC            ;SETUP WILD TRAP
15109  075154  177027                        LDCLD   (R7)+,AC0               ;*TEST INSTRUCTION
15110  075156  005201                        INC     R1                      ;
15111  075160  005201                        INC     R1                      ;
15112  075162  005201                        INC     R1                      ;
15113  075164  005201                        INC     R1                      ;
15114  075166  020127  000003                CMP     R1,#3                   ;VERIFY
15115  075172  001406                        BEQ     1$                      ;BRANCH IF GOOD
15116  075174  104003                        ERROR   .3                      ;FPP ERROR
15117                                                                         ;INSTRUCTION FAILED
15118  075176  000167  000006                JMP     1$                      ;JUMP OVER WILD TRAP
15119  075202  012600                  10$:    MOV     (SP)+,R0                ;SAVE PC
15120  075204  012605                          MOV     (SP)+,R5                        ;SAVE PS
15121  075206  104003                          ERROR   .3                      ;FPP ERROR
15122                                                                         ;WILD TRAP ON INSTRUCTION
15123  075210  012704  003264          1$:     MOV     #TAB6A,R4               ;POINT TO EXPECTED DATA
15124  075214  012701  003102                  MOV     #RECDST,R1              ;POINT TO DATA BUFFER
15125  075220  174011                          STD     AC0,(R1)                ;VERIFY DATA
15126  075222  004767  035444                  JSR     R7,DATVER               ;
15127  075226  005767  105626                  TST     COUNT
15128  075232  001401                          BEQ     2$                      ;BRANCH IF GOOD DATA
15129  075234  104003                          ERROR   .3                      ;FPP ERROR
15130                                                                         ;BAD DATA
15131  075236                          2$:
15132
15133
15134
15135                                ;
```

```
                                ;
                                ;- - -- -    -- ------ ---   ---
                                ;TEST LDCIF, LDCLF
                                ;
                                        ;
                                        ;
                                        ;
075236                          MLCF:

                                ;
                                ;1/INT=0
075236  004767  000500                  JSR     R7,LCFSUB               ;DO TEST
075242  000000  000000                  .WORD   0,0             ;FSRC
075246  000000  000000                  .WORD   0,0             ;RESULT
075252  000000                          .WORD   0               ; TEST FPS
075254  000004                          .WORD   4               ;RESULT FPS
                                ;2/INT=0,-1
075256  004767  000460                  JSR     R7,LCFSUB               ;DO TEST
075262  000000  177777                  .WORD   0,-1            ;FSRC
075266  000000  000000                  .WORD   0,0             ;RESULT
075272  007440                          .WORD   7440            ; TEST FPS
075274  007444                          .WORD   7444            ;RESULT FPS
                                ;3/LONG=0
075276  004767  000440                  JSR     R7,LCFSUB               ;DO TEST
075302  000000  000000                  .WORD   0,0             ;FSRC
075306  000000  000000                  .WORD   0,0             ;RESULT
075312  000100                          .WORD   100             ; TEST FPS
075314  000104                          .WORD   104             ;RESULT FPS
                                ;4/INT=40000
075316  004767  000420                  JSR     R7,LCFSUB               ;DO TEST
075322  040000  000000                  .WORD   40000,0         ;FSRC
075326  043600  000000                  .WORD   43600,0         ;RESULT
075332  000017                          .WORD   17              ; TEST FPS
075334  000000                          .WORD   0               ;RESULT FPS
                                ;5/LONG=1
075336  004767  000400                  JSR     R7,LCFSUB               ;DO TEST
075342  000000  000001                  .WORD   0,1             ;FSRC
075346  040200  000000                  .WORD   40200,0         ;RESULT
075352  000117                          .WORD   117             ; TEST FPS
075354  000100                          .WORD   100             ;RESULT FPS
                                ;6/INT=PATTERN
075356  004767  000360                  JSR     R7,LCFSUB               ;DO TEST
075362  000252  025252                  .WORD   252,25252       ;FSRC
075366  042052  000000                  .WORD   42052,0         ;RESULT
075372  000000                          .WORD   0               ; TEST FPS
075374  000000                          .WORD   0               ;RESULT FPS
                                ;7/INT=-40000
075376  004767  000340                  JSR     R7,LCFSUB               ;DO TEST
075402  140000  000000                  .WORD   -40000,0        ;FSRC
075406  143600  000000                  .WORD   143600,0                ;RESULT
075412  000007                          .WORD   7               ; TEST FPS
075414  000010                          .WORD   10              ;RESULT FPS
                                ;8/INT=-1
075416  004767  000320                  JSR     R7,LCFSUB               ;DO TEST
075422  177777  000000                  .WORD   -1,0            ;FSRC
075426  140200  000000                  .WORD   140200,0                ;RESULT
```

```
15192  075432  000007                          .WORD   7               ; TEST FPS
15193  075434  000010                          .WORD   10              ;RESULT FPS
15194                                  ;9/INT-PATTERN
15195  075436  004767  000300                  JSR     R7,LCFSUB               ;DO TEST
15196  075442  125252  125252                  .WORD   125252,125252           ;FSRC
15197  075446  143652  126000                  .WORD   143652,126000           ;RESULT
15198  075452  000007                          .WORD   7               ; TEST FPS
15199  075454  000010                          .WORD   10              ;RESULT FPS
15200                                  ;10/LONG=40000
15201  075456  004767  000260                  JSR     R7,LCFSUB               ;DO TEST
15202  075462  040000  000000                  .WORD   40000.0         ;FSRC
15203  075466  047600  000000                  .WORD   47600.0         ;RESULT
15204  075472  000117                          .WORD   117             ; TEST FPS
15205  075474  000100                          .WORD   100             ;RESULT FPS
15206                                  ;11/LONG=1
15207  075476  004767  000240                  JSR     R7,LCFSUB               ;DO TEST
15208  075502  000000  000001                  .WORD   0,1             ;FSRC
15209  075506  040200  000000                  .WORD   40200.0         ;RESULT
15210  075512  007557                          .WORD   7557            ; TEST FPS
15211  075514  007540                          .WORD   7540            ;RESULT FPS
15212                                  ;12/LONG=PATTERN
15213  075516  004767  000220                  JSR     R7,LCFSUB               ;DO TEST
15214  075522  000000  000252                  .WORD   0,252           ;FSRC
15215  075526  042052  000000                  .WORD   42052.0         ;RESULT
15216  075532  007557                          .WORD   7557            ; TEST FPS
15217  075534  007540                          .WORD   7540            ;RESULT FPS
15218                                  ;13/LONG = -40000
15219  075536  004767  000200                  JSR     R7,LCFSUB               ;DO TEST
15220  075542  140000  000000                  .WORD   -40000.0                ;FSRC
15221  075546  147600  000000                  .WORD   147600.0                ;RESULT
15222  075552  000107                          .WORD   107             ; TEST FPS
15223  075554  000110                          .WORD   110             ;RESULT FPS
15224                                  ;14/LONG=-1
15225  075556  004767  000160                  JSR     R7,LCFSUB               ;DO TEST
15226  075562  177777  177777                  .WORD   -1,-1           ;FSRC
15227  075566  140200  000000                  .WORD   140200.0                ;RESULT
15228  075572  007500                          .WORD   7500            ; TEST FPS
15229  075574  007510                          .WORD   7510            ;RESULT FPS
15230                                  ;15/LONG=PATTERN
15231  075576  004767  000140                  JSR     R7,LCFSUB               ;DO TEST
15232  075602  125252  125252                  .WORD   125252.125252           ;FSRC
15233  075606  147652  125253                  .WORD   147652.125253           ;RESULT
15234  075612  000105                          .WORD   105             ; TEST FPS
15235  075614  000110                          .WORD   110             ;RESULT FPS
15236                                  ;16/LONG=77777,177500
15237  075616  004767  000120                  JSR     R7,LCFSUB               ;DO TEST
15238  075622  077777  177500                  .WORD   77777.177500            ;FSRC
15239  075626  047777  177777                  .WORD   47777.177777            ;RESULT
15240  075632  000117                          .WORD   117             ; TEST FPS
15241  075634  000100                          .WORD   100             ;RESULT FPS
15242                                  ;17/LONG=40000,100
15243  075636  004767  000100                  JSR     R7,LCFSUB               ;DO TEST
15244  075642  040000  000100                  .WORD   40000,100               ;FSRC
15245  075646  047600  000001                  .WORD   47600,1         ;RESULT
15246  075652  007502                          .WORD   7502            ; TEST FPS
15247  075654  007500                          .WORD   7500            ;RESULT FPS
```

```
                                        ;18/LONG=40000,100   TRUNCATE
075656  004767  000060                          JSR     R7,LCFSUB                       ;DO TEST
075662  040000  000100                          .WORD   40000,100                       ;FSRC
075666  047600  000000                          .WORD   47600,0                 ;RESULT
075672  007557                                  .WORD   7557                    ; TEST FPS
075674  007540                                  .WORD   7540                    ;RESULT FPS
                                        ;19/INT= MOST NEGATIVE
075676  004767  000040                          JSR     R7,LCFSUB                       ;DO TEST
075702  100000  000000                          .WORD   100000,0                        ;FSRC
075706  144000  000000                          .WORD   144000,0                        ;RESULT
075712  000007                                  .WORD   7                       ; TEST FPS
075714  000010                                  .WORD   10                      ;RESULT FPS
                                        ;20/LONG= MOST NEGATIVE
075716  004767  000020                          JSR     R7,LCFSUB                       ;DO TEST
075722  100000  000000                          .WORD   100000,0                        ;FSRC
075726  150000  000000                          .WORD   150000,0                        ;RESULT
075732  000107                                  .WORD   107                     ; TEST FPS
075734  000110                                  .WORD   110                     ;RESULT FPS
                                        ;

075736  000167  000112                          JMP     HOP18                   ;GET OVER SUBROUTINE
                                        ;
                                        ;
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;LDCIF, LDCLF
                                        ;
                                        ;                       FSRC
                                        ;                       RESULT
                                        ;                       FPS BEFORE EXECUTION
                                        ;                       FPS AFTER EXECUTION
                                        ;
                                        ;NO TRAP CAN OCCUR
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;
075742  012602                          LCFSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
075744  012737  076042  000244                  MOV     #50$,@#FPVEC                    ;REDIRECT TRAP VECTOR
075752  012701  003102                          MOV     #RECDST,R1                      ;POINT TO RESULT AREA
075756  016200  000010                          MOV     10(R2),R0                       ;GET TEST FPS
075762  170100                                  LDFPS   R0                              ;LOAD TEST FPS
075764  010204                                  MOV     R2,R4                           ;POINT TO TEST DATA
                                        ;
075766  177014                          40$:    LDCIF   (R4),AC0                        ;*TEST INSTRUCTION (ACCORDING TO MODE)
                                        ;
                                        ;VERIFY STATUS
075770  170203                          2$:     STFPS   R3                              ;SAVE FPS
075772  012700  000200                          MOV     #200,R0                         ;SET FPP STATUS TO DOUBLE
075776  170100                                  LDFPS   R0                              ;
076000  174011                                  STD     AC0,(R1)                        ;SAVE TEST RESULT INTO RECDST
076002  016200  000012                          MOV     12(R2),R0                               ;GET EXPECTED STATUS
076006  020003                                  CMP     R0,R3                           ;VERIFY STATUS
076010  001401                                  BEQ     3$                              ;BRANCH IF GOOD
076012  104003                                  ERROR   +3                      ;FPP ERROR
                                                                                ;BAD FPS
076014  010204                          3$:     MOV     R2,R4                           ;POINT TO EXPECTED DATA
```

F6


```
15304  076016  062704  000004            ADD     #4,R4
15305  076022  004767  034626     4$:    JSR     R7,DATVFR                       ;VERIFY DATA
15306  076026  005767  105026            TST     COUNT
15307  076032  001401                    BEQ     5$                      ;BRANCH IF GOOD
15308  076034  104003                    ERROR   .+3                 ;FPP ERROR
15309                                                                 ;BAD ACO
15310  076036  000162  000014     5$:    JMP     14(R2)                  ;RETURN FROM TEST
15311                                    ;
15312                                    ;INSTRUCTION TRAPPED
15313  076042  012600                    50$:    MOV     (SP)+,R0                ;SAVE PC
15314  076044  012605                    MOV     (SP)+,R5                ;SAVE PS
15315  076046  104003                    ERROR   .+3                 ;FPP ERROR
15316                                                                 ;INSTRUCTION WASNT SUPPOSE TO TRAP
15317  076050  000167  177762            JMP     5$                      ;CONTINUE
15318  076054                    HOP18:
15319
15320                                    ;
15321                                    ;
15322                                    ;--------------------------------
15323                                    ;TEST LDCID, LDCLD
15324                                    ;
15325                                            ;
15326                                            ;
15327                                            ;
15328  076054                    MLCD:
15329
15330                                    ;
15331                                    ;1/LONG=0
15332  076054  004767  000264            JSR     R7,LCDSUB               ;DO TEST
15333  076060  000000  000000            .WORD   0,0                 ;FSRC
15334  076064  000000  000000  000000    .WORD   0,0,0,0             ;RESULT
15335  076072  000000
15336  076074  007313                    .WORD   7313                ; TEST FPS
15337  076076  007304                    .WORD   7304                ;RESULT FPS
15338                                    ;2/INT=0
15339  076100  004767  000240            JSR     R7,LCDSUB               ;DO TEST
15340  076104  000000  000001            .WORD   0,1                 ;FSRC
15341  076110  040200  000000  000000    .WORD   40200,0,0,0             ;RESULT
15342  076116  000000
15343  076120  007757                    .WORD   7757                ; TEST FPS
15344  076122  007740                    .WORD   7740                ;RESULT FPS
15345                                    ;3/INT=40000
15346  076124  004767  000214            JSR     R7,LCDSUB               ;DO TEST
15347  076130  040000  177777            .WORD   40000,-1                ;FSRC
15348  076134  043600  000000  000000    .WORD   43600,0,0,0             ;RESULT
15349  076142  000000
15350  076144  007617                    .WORD   7617                ; TEST FPS
15351  076146  007600                    .WORD   7600                ;RESULT FPS
15352                                    ;4/INT=-40000
15353  076150  004767  000170            JSR     R7,LCDSUB               ;DO TEST
15354  076154  140000  177777            .WORD   -40000,-1               ;FSRC
15355  076160  143600  000000  000000    .WORD   143600,0,0,0            ;RESULT
15356  076166  000000
15357  076170  007600                    .WORD   7600                ; TEST FPS
15358  076172  007610                    .WORD   7610                ;RESULT FPS
15359                                    ;5/LONG=40000
```

```
076174  004767  000144                  JSR     R7,LCDSUB                       ;DO TEST
076200  040000  000000                  .WORD   40000,0                 ;FSRC
076204  047600  000000  000000          .WORD   47600,0,0,0                     ;RESULT
076212  000000
076214  007757                          .WORD   7757                    ; TEST FPS
076216  007740                          .WORD   7740                    ;RESULT FPS
                                ;6/LONG=1
076220  004767  000120                  JSR     R7,LCDSUB                       ;DO TEST
076224  000000  000001                  .WORD   0,1                     ;FSRC
076230  040200  000000  000000          .WORD   40200,0,0,0                     ;RESULT
076236  000000
076240  000300                          .WORD   300                     ; TEST FPS
076242  000300                          .WORD   300                     ;RESULT FPS
                                ;7/LONG=-2
076244  004767  000074                  JSR     R7,LCDSUB                       ;DO TEST
076250  177777  177776                  .WORD   -1,-2                   ;FSRC
076254  140400  000000  000000          .WORD   140400,0,0,0                    ;RESULT
076262  000000
076264  007300                          .WORD   7300                    ; TEST FPS
076266  007310                          .WORD   7310                    ;RESULT FPS
                                ;8/INT=PATTERN
076270  004767  000050                  JSR     R7,LCDSUB                       ;DO TEST
076274  123456  176543                  .WORD   123456,176543                   ;FSRC
076300  143661  122000  000000          .WORD   143661,122000,0,0                       ;RESULT
076306  000000
076310  000200                          .WORD   200                     ; TEST FPS
076312  000210                          .WORD   210                     ;RESULT FPS
                                ;9/LONG=PATTERN
076314  004767  000024                  JSR     R7,LCDSUB                       ;DO TEST
076320  125252  125252                  .WORD   125252,125252                   ;FSRC
076324  147652  125252  126000          .WORD   147652,125252,126000,0                  ;RESULT
076332  000000
076334  000300                          .WORD   300                     ; TEST FPS
076336  000310                          .WORD   310                     ;RESULT FPS
                                ;

                                ;
076340  000167  000112                  JMP     HOP19                   ;GET OVER SUBROUTINE
                                ;
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;LDCID, LDCLD
                                ;
                                ;               FSRC
                                ;               RESULT
                                ;               FPS BEFORE EXECUTION
                                ;               FPS AFTER EXECUTION
                                ;
                                ;NO TRAP CAN OCCUR
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
076344  012602                  LCDSUB: MOV     (SP)+,R2                        ; RETURN ADDRESS TO USE AS POINTER
076346  012737  076444  000244          MOV     #50$,@#FPVEC                    ;REDIRECT TRAP VECTOR
076354  012701  003102                  MOV     #RECDST,R1                      ;POINT TO RESULT AREA
076360  016200  000014                  MOV     14(R2),R0                       ;GET TEST FPS
```

```
15416  076364  170100                  LDFPS   R0              ;LOAD TEST FPS
15417  076366  010204                  MOV     R2,R4           ;POINT TO TEST DATA
15418                                  ;
15419  076370  177014                  40$:    LDCID   (R4),AC0        ;*TEST INSTRUCTION (ACCORDING TO MODE)
15420                                  ;
15421                                  ;VERIFY STATUS
15422  076372  170203                  2$:     STFPS   R3              ;SAVE FPS
15423  076374  012700  000200                  MOV     #200,R0         ;SET FPP STATUS TO DOUBLE
15424  076400  170100                  LDFPS   R0                      ;
15425  076402  174011                  STD     AC0,(R1)                ;SAVE TEST RESULT INTO RECDST
15426  076404  016200  000016                  MOV     16(R2),R0               ;GET EXPECTED STATUS
15427  076410  020003                  CMP     R0,R3           ;VERIFY STATUS
15428  076412  001401                  BEQ     3$              ;BRANCH IF GOOD
15429  076414  104003                  ERROR   .3      ;FPP ERROR
15430                                                  ;BAD FPS
15431  076416  010204                  3$:     MOV     R2,R4           ;POINT TO EXPECTED DATA
15432  076420  062704  000004                  ADD     #4,R4
15433  076424  004767  034242          4$:     JSR     R7,DATVER               ;VERIFY DATA
15434  076430  005767  104424                  TST     COUNT
15435  076434  001401                  BEQ     5$              ;BRANCH IF GOOD
15436  076436  104003                  ERROR   .3      ;FPP ERROR
15437                                                  ;BAD AC0
15438  076440  000162  000020          5$:     JMP     20(R2)          ;RETURN FROM TEST
15439                                  ;
15440                                  ;INSTRUCTION TRAPPED
15441  076444  012600                  50$:    MOV     (SP)+,R0        ;SAVE PC
15442  076446  012605                  MOV     (SP)+,R5        ;SAVE PS
15443  076450  104003                  ERROR   .3      ;FPP ERROR
15444                                                  ;INSTRUCTION WASNT SUPPOSE TO TRAP
15445  076452  000167  177762                  JMP     5$              ;CONTINUE
15446                                  ;
15447  076456                          MOP19:
15448
15449                                  ;
15450                                  ;
15451                                  ;---------------------------------
15452                                  ;TEST LDEXP
15453                                  ;DOUBLE
15454                                  ;
15455                                          ;
15456                                          ;
15457                                          ;
15458  076456                          MLXP:
15459
15460                                  ;
15461                                  ;1/EXP=10 - AC=NEG
15462  076456  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPTS
15463  076462  004767  001140                  JSR     R7,LXPSUB               ;DO TEST
15464  076466  123456  067012  025252          .WORD   123456,67012,25252,171717       ;AC0
15465  076474  171717
15466  076476  000010                  .WORD   10                      ;EXP
15467  076500  142056  067012  025252          .WORD   142056,67012,25252,171717               ;RESULT
15468  076506  171717
15469  076510  007757                  .WORD   7757            ; TEST FPS
15470  076512  007750                  .WORD   7750            ;RESULT FPS
15471                                  ;2/EXP=177 - AC0=POS
```

```
15472  076514  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPTS
15473  076520  004767  001102                  JSR     R7,LXPSUB               ;DO TEST
15474  076524  023456  070123  100000          .WORD   23456,70123,100000,1            ;ACO
15475  076532  000001
15476  076534  000177                          .WORD   177             ;EXP
15477  076536  077656  070123  100000          .WORD   77656,70123,100000,1            ;RESULT
15478  076544  000001
15479  076546  007700                          .WORD   7700            ; TEST FPS
15480  076550  007700                          .WORD   7700            ;RESULT FPS
15481                          ;3/EXP=56
15482  076552  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPTS
15483  076556  004767  001044                  JSR     R7,LXPSUB               ;DO TEST
15484  076562  055555  044444  033333          .WORD   55555,44444,33333,22222         ;ACO
15485  076570  022222
15486  076572  000056                          .WORD   56              ;EXP
15487  076574  053555  044444  033333          .WORD   53555,44444,33333,22222 ;RESULT
15488  076602  022222
15489  076604  007757                          .WORD   7757            ; TEST FPS
15490  076606  007740                          .WORD   7740            ;RESULT FPS
15491                          ;4/EXP=-151, ACO=UV
15492  076610  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPTS
15493  076614  004767  001006                  JSR     R7,LXPSUB               ;DO TEST
15494  076620  100077  177777  177777          .WORD   100077,-1,-1,-2         ;ACO
15495  076626  177776
15496  076630  177623                          .WORD   -155            ;EXP
15497  076632  104677  177777  177777          .WORD   104677,-1,-1,-2         ;RESULT
15498  076640  177776
15499  076642  007757                          .WORD   7757            ; TEST FPS
15500  076644  007750                          .WORD   7750            ;RESULT FPS
15501                          ;5/EXP=-177
15502  076646  005037  002776                  CLR     @#FLAG                  ;NO INTERRUPTS
15503  076652  004767  000750                  JSR     R7,LXPSUB               ;DO TEST
15504  076656  000177  177777  177777          .WORD   177,-1,-1,-2            ;ACO
15505  076664  177776
15506  076666  177601                          .WORD   -177            ;EXP
15507  076670  000377  177777  177777          .WORD   377,-1,-1,-2            ;RESULT
15508  076676  177776
15509  076700  007700                          .WORD   7700            ; TEST FPS
15510  076702  007700                          .WORD   7700            ;RESULT FPS
15511                          ;6/EXP=-200, UNDERFLOW
15512  076704  012737  000001  002776          MOV     #1,@#FLAG               ; INTERRUPTS
15513  076712  004767  000710                  JSR     R7,LXPSUB               ;DO TEST
15514  076716  030131  032334  035363          .WORD   30131,32334,35363,73031         ;ACO
15515  076724  073031
15516  076726  177600                          .WORD   -200            ;EXP
15517  076730  000131  032334  035363          .WORD   131,32334,35363,73031           ;RESULT
15518  076736  073031
15519  076740  007740                          .WORD   7740            ; TEST FPS
15520  076742  107744                          .WORD   107744          ;RESULT FPS
15521  076744  000012                          .WORD   12              ;FEC
15522                          ;7/EXP=LARGEST NEGATIVE
15523  076746  012737  000001  002776          MOV     #1,@#FLAG               ;EXPECT  INTERRUPTS
15524  076754  004767  000646                  JSR     R7,LXPSUB               ;DO TEST
15525  076760  000000  000123  000456          .WORD   0,123,456,1             ;ACO
15526  076766  000001
15527  076770  100000                          .WORD   100000          ;EXP
```

J6


```
15528  076772  040000  000123  000456          .WORD   40000,123,456,1         ;RESULT
15529  077000  000001
15530  077002  002200                          .WORD   2200            ; TEST FPS
15531  077004  102200                          .WORD   102200          ;RESULT FPS
15532  077006  000012                          .WORD   12              ;FEC
15533                          ;8/EXP=-200, NEG. ACO
15534  077010  012737  000001  002776          MOV     #1,@#FLAG               ; INTERRUPTS
15535  077016  004767  000604                  JSR     R7,LXPSUB               ;DO TEST
15536  077022  111111  100000  100000          .WORD   111111,100000,100000,-1         ;ACO
15537  077030  177777
15538  077032  177600                          .WORD   -200            ;EXP
15539  077034  100111  100000  100000          .WORD   100111,100000,100000,-1         ;RESULT
15540  077042  177777
15541  077044  002217                          .WORD   2217            ; TEST FPS
15542  077046  102214                          .WORD   102214          ;RESULT FPS
15543  077050  000012                          .WORD   12              ;FEC
15544                          ;9/EXP=-1743, FIU=0
15545  077052  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPTS
15546  077060  004767  000542                  JSR     R7,LXPSUB               ;DO TEST
15547  077064  123456  012346  012346          .WORD   123456,12346,12346,123          ;ACO
15548  077072  000123
15549  077074  176035                          .WORD   -1743           ;EXP
15550  077076  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
15551  077104  000000
15552  077106  005700                          .WORD   5700            ; TEST FPS
15553  077110  005704                          .WORD   5704            ;RESULT FPS
15554  077112  000012                          .WORD   12              ;FEC
15555                          ;10/EXP =-16616, FID=1
15556  077114  012737  000002  002776          MOV     #2,@#FLAG               ;NO INTERRUPTS
15557  077122  004767  000500                  JSR     R7,LXPSUB               ;DO TEST
15558  077126  000377  123456  065432          .WORD   377,123456,65432,1              ;ACO
15559  077134  000001
15560  077136  161162                          .WORD   -16616          ;EXP
15561  077140  074577  123456  065432          .WORD   74577,123456,65432,1            ;RESULT
15562  077146  000001
15563  077150  047700                          .WORD   47700           ; TEST FPS
15564  077152  147700                          .WORD   147700          ;RESULT FPS
15565  077154  000012                          .WORD   12              ;FEC
15566                          ;11/EXP=177, ACO=UNDEFINED VARIABLE
15567  077156  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
15568  077162  004767  000440                  JSR     R7,LXPSUB               ;DO TEST
15569  077166  100177  177777  177777          .WORD   100177,-1,-1,-1         ;ACO
15570  077174  177777
15571  077176  000177                          .WORD   177             ;EXP
15572  077200  177777  177777  177777          .WORD   -1,-1,-1,-1             ;RESULT
15573  077206  177777
15574  077210  007700                          .WORD   7700            ; TEST FPS
15575  077212  007710                          .WORD   7710            ;RESULT FPS
15576                          ;12/EXP=150 ACO=POS
15577  077214  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
15578  077220  004767  000402                  JSR     R7,LXPSUB               ;DO TEST
15579  077224  000200  000100  000200          .WORD   200,100,200,300         ;ACO
15580  077232  000300
15581  077234  000150                          .WORD   150             ;EXP
15582  077236  072000  000100  000200          .WORD   72000,100,200,300               ;RESULT
15583  077244  000300
```

```
15584  077246  007717                          .WORD   7717            ; TEST FPS
15585  077250  007700                          .WORD   7700            ;RESULT FPS
15586                          ;13/EXP=200, AC0=NEG
15587  077252  012737  000001  002776          MOV     #1,@#FLAG
15588  077260  004767  000342                  JSR     R7,LXPSUB               ;DO TEST
15589  077264  177777  177777  177777          .WORD   -1,-1,-1,-1             ;AC0
15590  077272  177777
15591  077274  000200                          .WORD   200             ;EXP
15592  077276  100177  177777  177777          .WORD   100177,-1,-1,-1         ;RESULT
15593  077304  177777
15594  077306  007705                          .WORD   7705            ; TEST FPS
15595  077310  107716                          .WORD   107716          ;RESULT FPS
15596  077312  000010                          .WORD   10              ;FEC
15597                          ;14/EXP=400, FID
15598  077314  012737  000002  002776          MOV     #2,@#FLAG               ;INTERRUPT
15599  077322  004767  000300                  JSR     R7,LXPSUB               ;DO TEST
15600  077326  000555  177777  177776          .WORD   555,-1,-2,-3            ;AC0
15601  077334  177775
15602  077336  000400                          .WORD   400                     ;EXP
15603  077340  040155  177777  177776          .WORD   40155,-1,-2,-3          ;RESULT
15604  077346  177775
15605  077350  047700                          .WORD   47700           ; TEST FPS
15606  077352  147702                          .WORD   147702          ;RESULT FPS
15607  077354  000010                          .WORD   10              ;FEC
15608                          ;15/EXP=11011 FIU=0
15609  077356  012737  000000  002776          MOV     #0,@#FLAG               ;NO INTERRUPT
15610  077364  004767  000236                  JSR     R7,LXPSUB               ;DO TEST
15611  077370  177773  177777  177776          .WORD   177773,-1,-2,-3         ;AC0
15612  077376  177775
15613  077400  011011                          .WORD   11011           ;EXP
15614  077402  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
15615  077410  000000
15616  077412  006700                          .WORD   6700            ; TEST FPS
15617  077414  006706                          .WORD   6706            ;RESULT FPS
15618                          ;16/EXP=LARGEST POSITIVE
15619  077416  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
15620  077424  004767  000176                  JSR     R7,LXPSUB               ;DO TEST
15621  077430  123456  000100  000100          .WORD   123456,100,100,200              ;AC0
15622  077436  000200
15623  077440  077777                          .WORD   77777           ;EXP
15624  077442  137656  000100  000100          .WORD   137656,100,100,200              ;RESULT
15625  077450  000200
15626  077452  007740                          .WORD   7740            ; TEST FPS
15627  077454  107752                          .WORD   107752          ;RESULT FPS
15628  077456  000010                          .WORD   10              ;FEC
15629                          ;17/FLOATING
15630  077460  005037  002776                  CLR     @#FLAG          ;NO INTERRUPT
15631  077464  004767  000136                  JSR     R7,LXPSUB               ;DO TEST
15632  077470  123456  023465  000555          .WORD   123456,23465,555,444            ;AC0
15633  077476  000444
15634  077500  000050                          .WORD   50              ;EXP
15635  077502  152056  023465  000555          .WORD   152056,23465,555,444            ;RESULT
15636  077510  000444
15637  077512  007500                          .WORD   7500            ; TEST FPS
15638  077514  007510                          .WORD   7510            ;RESULT FPS
15639                          ;18/FLOATING UNDERFLOW
```

```
 15640  077516  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
 15641  077524  004767  000076                  JSR     R7,LXPSUB               ;DO TEST
 15642  077530  000333  000444  000555          .WORD   333,444,555,666         ;ACO
 15643  077536  000666
 15644  077540  177600                          .WORD   -200                ;EXP
 15645  077542  000133  000444  000555          .WORD   133,444,555,666         ;RESULT
 15646  077550  000666
 15647  077552  007500                          .WORD   7500                ; TEST FPS
 15648  077554  107504                          .WORD   107504              ;RESULT FPS
 15649  077556  000012                          .WORD   12                  ;FEC
 15650                          ;19/FLOATING OVERFLOW
 15651  077560  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
 15652  077566  004767  000034                  JSR     R7,LXPSUB               ;DO TEST
 15653  077572  012346  000123  000345          .WORD   12346,123,345,456               ;ACO
 15654  077600  000456
 15655  077602  000400                          .WORD   400                 ;EXP
 15656  077604  040146  000123  000345          .WORD   40146,123,345,456               ;RESULT
 15657  077612  000456
 15658  077614  007400                          .WORD   7400                ; TEST FPS
 15659  077616  107402                          .WORD   107402              ;RESULT FPS
 15660  077620  000010                          .WORD   10                  ;FEC
 15661                          ;
 15662
 15663  077622  000167  000220                  JMP     @OP20                   ;GET OVER SUBROUTINE
 15664                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 15665                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 15666                          ;LDEXP
 15667                          ;               ACO
 15668                          ;               EXPONENT
 15669                          ;               RESULT
 15670                          ;               FPS BEFORE EXECUTION
 15671                          ;               FPS AFTER EXECUTION
 15672                          ;               (FEC)
 15673                          ;
 15674                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 15675                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
 15676                          ;
 15677  077626  012602                  LXPSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
 15678  077630  012737  077712  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
 15679  077636  012701  003102                  MOV     #RECDST,R1              ;POINT TO RESULT AREA
 15680  077642  012700  000200                  MOV     #200,R0                 ;SET FPS TO DOUBLE
 15681  077646  170100                          LDFPS   R0                      ;
 15682  077650  010204                          MOV     R2,R4                   ;POINT TO ACO DATA
 15683  077652  172414                          LDD     (R4),AC0                        ;LOAD ACO
 15684  077654  016200  000022                  MOV     22(R2),R0               ;GET TEST FPS
 15685  077660  170100                          LDFPS   R0                      ;LOAD TEST FPS
 15686  077662  016204  000010                  MOV     10(R2),R4               ;POINT TO TEST DATA
 15687                          ;
 15688  077666  176404                  40$:    LDEXP   R4,AC0                  ;*TEST INSTRUCTION (ACCORDING TO MODE)
 15689  077670  170327                  1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA TRAP.
 15690  077672  000000                          .WORD   0                       ;STORE STATUS HERE
 15691                          ;
 15692                          ;
 15693                          ;INSTRUCTION DIDNT TRAP
 15694  077674  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
 15695  077702  001420                          BEQ     2$                      ;BRANCH IF GOOD
```

M6


```
15696  077704  104003                          ERROR   +3              ;FPP ERROR
15697                                                                   ;INSTRUCTION SHOULD HAVE TRAPPED
15698  077706  000167  000032                   JMP     2$                      ;REJOIN CODE
15699                                  ;
15700                                  ;INSTRUCTION TRAPPED
15701  077712  032737  000001  002776  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
15702  077720  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
15703  077722  104003                          ERROR   +3              ;FPP ERROR
15704                                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
15705  077724  000167  000014                   JMP     2$                      ;REJOIN CODE
15706  077730  012604                  51$:     MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
15707  077732  005726                          TST     (SP)+                   ;CLEAN UP STACK
15708  077734  022704  077670                   CMP     #1$,R4                  ;
15709  077740  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
15710  077742  104003                          ERROR   +3              ;FPP ERROR
15711                                                                   ;PC WAS INCORRECT
15712                                  ;
15713                                  ;COMMON CODE FOR TRAP AND NO TRAP
15714                                  ;VERIFY STATUS
15715  077744  170203                  2$:     STFPS   R3                      ;SAVE FPS
15716  077746  012700  000200                   MOV     #200,R0                 ;SETUP FPS
15717  077752  170100                          LDFPS   R0                      ;FPS=200
15718  077754  174011                          STD     AC0,(R1)                ;GET RESULT
15719  077756  016200  000024                   MOV     24(R2),R0                       ;GET EXPECTED STATUS
15720  077762  020003                          CMP     R0,R3                   ;VERIFY STATUS
15721  077764  001401                          BEQ     3$                      ;BRANCH IF GOOD
15722  077766  104003                          ERROR   +3              ;FPP ERROR
15723                                                                   ;BAD FPS
15724  077770  010204                  3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
15725  077772  062704  000012                   ADD     #12,R4
15726  077776  004767  032670          4$:     JSR     R7,DATVER                       ;VERIFY DATA
15727  100002  005767  103052                  TST     COUNT
15728  100006  001401                          BEQ     5$                      ;BRANCH IF GOOD
15729  100010  104003                          ERROR   +3              ;FPP ERROR
15730                                                                   ;BAD AC0
15731  100012  005737  002776          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
15732  100016  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
15733  100020  000162  000026                   JMP     26(R2)                  ;RETURN FROM TEST
15734                                  ;VERIFY FEC
15735  100024  012704  003062          7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
15736  100030  170314                          STST    (R4)                    ;SAVE FEC
15737  100032  021462  000026                   CMP     (R4),26(R2)                     ;VERIFY FEC FOR OVERFLOW
15738  100036  001401                          BEQ     8$                      ;BRANCH IF GOOD
15739  100040  104003                          ERROR   +3              ;FPP ERROR
15740                                                                   ;BAD FEC
15741  100042  000162  000030          8$:     JMP     30(R2)          ;RETURN FROM TEST
15742                                  ;
15743  100046                          MOP20:
15744
15745                                  ;
15746                                  ;
15747                                  ;---------------------------------
15748                                  ;TEST STCDI, STCDL
15749                                  ;
15750                                          ;
15751                                          ;
```

```
 15752                                          ;
 15753  100046                                  MSCD:
 15754
 15755                                          ;
 15756                                          ;1/ACO=0, INT
 15757  100046  005037  002776                          CLR     @#FLAG          ;NO INTERRUPTS
 15758  100052  004767  000610                          JSR     R7,SCDSUB               ;DO TEST
 15759  100056  000177  000000  000000                  .WORD   0177,0,0,0              ;ACO
 15760  100064  000000
 15761  100066  000000  177777                          .WORD   0,-1            ;RESULT
 15762  100072  007640                                  .WORD   7640            ; TEST FPS
 15763  100074  007644                                  .WORD   7644            ;RESULT FPS
 15764                                          ;2/ACO=-0, LONG
 15765  100076  005037  002776                          CLR     @#FLAG          ;INTERRUPT
 15766  100102  004767  000560                          JSR     R7,SCDSUB               ;DO TEST
 15767  100106  100177  177777  177777                  .WORD   100177,-1,-1,-1         ;ACO
 15768  100114  177777
 15769  100116  000000  000000                          .WORD   0,0             ;RESULT
 15770  100122  007700                                  .WORD   7700            ; TEST FPS
 15771  100124  007704                                  .WORD   7704            ;RESULT FPS
 15772                                          ;3/EXP=100, LONG
 15773  100126  005037  002776                          CLR     @#FLAG                  ;NO INTERRUPT
 15774  100132  004767  000530                          JSR     R7,SCDSUB               ;DO TEST
 15775  100136  020000  000000  000000                  .WORD   20000,0,0,0             ;ACO
 15776  100144  000000
 15777  100146  000000  000000                          .WORD   0,0             ;RESULT
 15778  100152  000300                                  .WORD   300             ; TEST FPS
 15779  100154  000304                                  .WORD   304             ;RESULT FPS
 15780                                          ;4/EXP=200, BAISED 0, INT, ROUND
 15781  100156  005037  002776                          CLR     @#FLAG          ;INTERRUPT
 15782  100162  004767  000500                          JSR     R7,SCDSUB               ;DO TEST
 15783  100166  140177  177777  000001                  .WORD   140177,177777,1,1               ;ACO
 15784  100174  000001
 15785  100176  000000  000000                          .WORD   0,0             ;RESULT
 15786  100202  007700                                  .WORD   7700            ; TEST FPS
 15787  100204  007704                                  .WORD   7704            ;RESULT FPS
 15788                                          ;5/LONG
 15789  100206  005037  002776                          CLR     @#FLAG          ;INTERRUPT
 15790  100212  004767  000450                          JSR     R7,SCDSUB               ;DO TEST
 15791  100216  047667  075757  157737                  .WORD   47667,75757,157737,167773               ;ACO
 15792  100224  167773
 15793  100226  055675  173757                          .WORD   55675,173757            ;RESULT
 15794  100232  007717                                  .WORD   7717            ; TEST FPS
 15795  100234  007700                                  .WORD   7700            ;RESULT FPS
 15796                                          ;6/LONG, EXP=2**32
 15797  100236  005037  002776                          CLR     @#FLAG                  ;NO INTERRUPT
 15798  100242  004767  000420                          JSR     R7,SCDSUB               ;DO TEST
 15799  100246  046400  000000  000000                  .WORD   46400,0,0,0             ;ACO
 15800  100254  000000
 15801  100256  001000  000000                          .WORD   1000,0          ;RESULT
 15802  100262  007700                                  .WORD   7700            ; TEST FPS
 15803  100264  007700                                  .WORD   7700            ;RESULT FPS
 15804                                          ;7/LONG, EXP>2**32
 15805  100266  012737  000001  002776                  MOV     #1,@#FLAG               ;INTERRUPT
 15806  100274  004767  000366                          JSR     R7,SCDSUB               ;DO TEST
 15807  100300  077607  000000  000000                  .WORD   77607,0,0,0             ;ACO
```

```
15808  100306  000000
15809  100310  000000  000000                .WORD   0,0             ;RESULT
15810  100314  007700                        .WORD   7700            ; TEST FPS
15811  100316  107705                        .WORD   107705          ;RESULT FPS
15812                          ;8/INT, EXP=2**15
15813  100320  005037  002776                CLR     @#FLAG          ;NO INTERRUPTS
15814  100324  004767  000336                JSR     R7,SCDSUB               ;DO TEST
15815  100330  043200  000000  000000        .WORD   43200,0,0,0             ;ACO
15816  100336  000000
15817  100340  010000  177777                .WORD   10000,-1                ;RESULT
15818  100344  007600                        .WORD   7600            ; TEST FPS
15819  100346  007600                        .WORD   7600            ;RESULT FPS
15820                          ;9/INT, EXP>2**15
15821  100350  012737  000001  002776        MOV     #1,@#FLAG               ;INTERRUPT
15822  100356  004767  000304                JSR     R7,SCDSUB               ;DO TEST
15823  100362  077777  177777  177777        .WORD   77777,-1,-1,-1          ;ACO
15824  100370  177777
15825  100372  000000  177777                .WORD   0,-1            ;RESULT
15826  100376  007600                        .WORD   7600            ; TEST FPS
15827  100400  107605                        .WORD   107605          ;RESULT FPS
15828                          ;10/INT, EXP>2**15, FID
15829  100402  012737  000000  002776        MOV     #0,@#FLAG               ;NO INTERRUPT
15830  100410  004767  000252                JSR     R7,SCDSUB               ;DO TEST
15831  100414  043300  000000  000000        .WORD   43300,0,0,0             ;ACO
15832  100422  000000
15833  100424  000000  014000                .WORD   0,14000         ;RESULT
15834  100430  047700                        .WORD   47700           ; TEST FPS
15835  100432  047700                        .WORD   47700           ;RESULT FPS
15836                          ;11/INT, EXP>2**15, FIC=0
15837  100434  012737  000000  002776        MOV     #0,@#FLAG               ;NO INTERRUPT
15838  100442  004767  000220                JSR     R7,SCDSUB               ;DO TEST
15839  100446  143300  177777  177777        .WORD   143300,-1,-1,-1         ;ACO
15840  100454  177777
15841  100456  177777  163741                .WORD   -1,163741               ;RESULT
15842  100462  007300                        .WORD   7300            ; TEST FPS
15843  100464  007310                        .WORD   7310            ;RESULT FPS
15844                          ;12/LONG, EXP>2**32, FID
15845  100466  012737  000002  002776        MOV     #2,@#FLAG               ;INTERRUPT
15846  100474  004767  000166                JSR     R7,SCDSUB               ;DO TEST
15847  100500  050100  000000  000000        .WORD   50100,0,0,0             ;ACO
15848  100506  000000
15849  100510  000000  000000                .WORD   0,0             ;RESULT
15850  100514  047700                        .WORD   47700           ; TEST FPS
15851  100516  147705                        .WORD   147705          ;RESULT FPS
15852                          ;13/LONG, EXP>2**32, FIC=0
15853  100520  012737  000000  002776        MOV     #0,@#FLAG               ;NO INTERRUPT
15854  100526  004767  000134                JSR     R7,SCDSUB               ;DO TEST
15855  100532  050377  177777  177777        .WORD   50377,-1,-1,-1          ;ACO
15856  100540  177777
15857  100542  000000  000000                .WORD   0,0             ;RESULT
15858  100546  007300                        .WORD   7300            ; TEST FPS
15859  100550  007305                        .WORD   7305            ;RESULT FPS
15860                          ;14/LONG, EXP<0
15861  100552  005037  002776                CLR     @#FLAG          ;NO INTERRUPTS
15862  100556  004767  000104                JSR     R7,SCDSUB               ;DO TEST
15863  100562  100200  177777  177777        .WORD   100200,-1,-1,-1         ;ACO
```

```
100570  177777
100572  000000  000000                  .WORD   0,0             ;RESULT
100576  007757                          .WORD   7757            ; TEST FPS
100600  007744                          .WORD   7744            ;RESULT FPS
                                ;15/INT, EXP<0
100602  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
100606  004767  000054                  JSR     R7,SCDSUB               ;DO TEST
100612  037700  177777  177777          .WORD   37700,-1,-1,-2          ;ACO
100620  177776
100622  000000  177777                  .WORD   0,-1            ;RESULT
100626  007600                          .WORD   7600            ; TEST FPS
100630  007604                          .WORD   7604            ;RESULT FPS
                                ;16/INT, EXP=10
100632  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
100636  004767  000024                  JSR     R7,SCDSUB               ;DO TEST
100642  004377  177777  177777          .WORD   4377,-1,-1,-1           ;ACO
100650  177777
100652  000000  177777                  .WORD   0,-1            ;RESULT
100656  007600                          .WORD   7600            ; TEST FPS
100660  007604                          .WORD   7604            ;RESULT FPS
                                ;

100662  000167  000214                  JMP     HOP21           ;GET OVER SUBROUTINE
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;STCDI, STCDL, STCFI, STCFL
                                ;
                                ;               ACO
                                ;               RESULT
                                ;               FPS BEFORE EXECUTION
                                ;               FPS AFTER EXECUTION
                                ;               (FEC)
                                ;
                                ;TRAP ON CONVERSION FAILURE
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
100666  012602                  SCDSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
100670  012737  100756  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
100676  012701  003104                  MOV     #RECDST+2,R1            ;POINT TO RESULT AREA
100702  012711  177777                  MOV     #-1,(R1)                ;PRELOAD RECEIVE DATA BUFFER
100706  012741  177777                  MOV     #-1,-(R1)               ;
100712  012700  000200                  MOV     #200,R0                 ;SET FPS TO DOUBLE
100716  170100                          LDFPS   R0                      ;
100720  010204                          MOV     R2,R4                   ;POINT TO ACO DATA
100722  172414                          LDD     (R4),ACO                        ;LOAD ACO
100724  016200  000014                  MOV     14(R2),R0               ;GET TEST FPS
100730  170100                          LDFPS   R0                      ;LOAD TEST FPS
                                ;
100732  175411                  40$:    STCDI   ACO,(R1)                ;*TEST INSTRUCTION(ACCORDING TO MODE)
100734  170327                  1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA 'RAP.
100736  000000                          .WORD   0                       ;STORE STATUS HERE.
                                ;
                                ;
                                ;INSTRUCTION DIDNT TRAP
100740  032737  000001  002776          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
```

```
15920  100746  001420                          BEQ     2$                      ;BRANCH IF GOOD
15921  100750  104003                          ERROR   .3                  ;FPP ERROR
15922                                                                       ;INSTRUCTION SHOULD HAVE TRAPPED
15923  100752  000167  000032                  JMP     2$                      ;REJOIN CODE
15924                                  ;
15925                                  ;INSTRUCTION TRAPPED
15926  100756  032737  000001  002776 50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
15927  100764  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
15928  100766  104003                          ERROR   .3                  ;FPP ERROR
15929                                                                       ;INSTRUCTION WASNT SUPPOSE TO TRAP
15930  100770  000167  000014                  JMP     2$                      ;REJOIN CODE
15931  100774  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
15932  100776  005726                          TST     (SP)+                   ;CLEAN UP STACK
15933  101000  022704  100734                  CMP     #1$,R4                  ;
15934  101004  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
15935  101006  104003                          ERROR   .3                  ;FPP ERROR
15936                                                                       ;PC WAS INCORRECT
15937                                  ;
15938                                  ;COMMON CODE FOR TRAP AND NO TRAP
15939                                  ;VERIFY STATUS
15940  101010  170203                  2$:     STFPS   R3                      ;SAVE FPS
15941  101012  016200  000016                  MOV     16(R2),R0                       ;GET EXPECTED STATUS
15942  101016  020003                          CMP     R0,R3                   ;VERIFY STATUS
15943  101020  001401                          BEQ     3$                      ;BRANCH IF GOOD
15944  101022  104003                          ERROR   .3                  ;FPP ERROR
15945                                                                       ;BAD FPS
15946  101024  010204                  3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
15947  101026  062704  000010                  ADD     #10,R4
15948  101032  004767  031616          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
15949  101036  005767  102016                  TST     COUNT
15950  101042  001401                          BEQ     5$                      ;BRANCH IF GOOD
15951  101044  104003                          ERROR   .3                  ;FPP ERROR
15952                                                                       ;BAD ACO
15953  101046  005737  002776          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
15954  101052  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
15955  101054  000162  000020                  JMP     20(R2)                  ;RETURN FROM TEST
15956                                  ;VERIFY FEC
15957  101060  012704  003062          7$:     MOV     @#RECFEC,R4             ;POINT TO FEC AREA
15958  101064  170314                          STST    (R4)                    ;SAVE FEC
15959  101066  021427  000006                  CMP     (R4),#6                 ;VERIFY FEC FOR OVERFLOW
15960  101072  001401                          BEQ     8$                      ;BRANCH IF GOOD
15961  101074  104003                          ERROR   .3                  ;FPP ERROR
15962                                                                       ;BAD FEC
15963  101076  000162  000020          8$:     JMP     20(R2)          ;RETURN FROM TEST
15964                                  ;
15965  101102                          MOP21:
15966                                  ;
15967                                  ;
15968                                  ;
15969                                  ;---------------------------------
15970                                  ;TEST STCFI, STCFL
15971                                  ;
15972                                          ;
15973                                          ;
15974                                          ;
15975  101102                          MSCF:
```

```
15976
15977                                   ;
15978                                   ;1/LONG EXP =30
15979  101102  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
15980  101106  004767  177554                  JSR     R7,SCDSUB               ;DO TEST
15981  101112  044541  052525  177777          .WORD   44541,52525, 1,-1               ;ACO
15982  101120  177777
15983  101122  000003  102525                  .WORD   3,102525                ;RESULT
15984  101126  007517                          .WORD   7517            ; TEST FPS
15985  101130  007500                          .WORD   7500            ;RESULT FPS
15986                                   ;2/INT, EXP<0
15987  101132  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
15988  101136  004767  177524                  JSR     R7,SCDSUB               ;DO TEST
15989  101142  002300  177777  177777          .WORD   2300,-1,-1,-1           ;ACO
15990  101150  177777
15991  101152  000000  177777                  .WORD   0,-1            ;RESULT
15992  101156  007400                          .WORD   7400            ; TEST FPS
15993  101160  007404                          .WORD   7404            ;RESULT FPS
15994                                   ;3/LONG, EXP
15995  101162  012737  000001  002776          MOV     #1,@#FLAG               ;INTERRUPT
15996  101170  004767  177472                  JSR     R7,SCDSUB               ;DO TEST
15997  101174  070000  177777  177777          .WORD   70000,-1,-1,-1          ;ACO
15998  101202  177777
15999  101204  000000  000000                  .WORD   0,0             ;RESULT
16000  101210  007540                          .WORD   7540            ; TEST FPS
16001  101212  107545                          .WORD   107545          ;RESULT FPS
16002                                   ;4/INT,EXP=5, FIC=0, FID=1
16003  101214  005037  002776                  CLR     @#FLAG          ;NO INTERRUPTS
16004  101220  004767  177442                  JSR     R7,SCDSUB               ;DO TEST
16005  101224  052000  000000  177777          .WORD   52000,0,-1,-1           ;ACO
16006  101232  177777
16007  101234  000000  177777                  .WORD   0,-1            ;RESULT
16008  101240  047000                          .WORD   47000           ; TEST FPS
16009  101242  047005                          .WORD   47005           ;RESULT FPS
16010                                   ;
16011
16012
16013                                   ;
16014                                   ;
16015                                   ;---------------------------------
16016                                   ;TEST STEXP
16017                                   ;
16018                                           ;
16019                                           ;
16020                                           ;
16021  101244                           MSXP:
16022
16023                                   ;
16024                                   ;1/EXP=100
16025  101244  004767  000154                  JSR     R7,SXPSUB               ;DO TEST
16026  101250  020000  000000  000000          .WORD   20000,0,0,0             ;ACO
16027  101256  000000
16028  101260  177700                          .WORD   -100            ;RESULT
16029  101262  007740                          .WORD   7740            ; TEST FPS
16030  101264  007750                          .WORD   7750            ;RESULT FPS
16031                                   ;2/EXP=201 FLOAT, NEG
```

```
16032  101266  004767  000132                  JSR     R7,SXPSUB               ;DO TEST
16033  101272  140377  177777  177777          .WORD   140377, 1,-1,0          ;AC0
16034  101300  000000
16035  101302  000001                          .WORD   1               ;RESULT
16036  101304  007500                          .WORD   7500            ; TEST FPS
16037  101306  007500                          .WORD   7500            ;RESULT FPS
16038                                  ;3/EXP=-177
16039  101310  004767  000110                  JSR     R7,SXPSUB               ;DO TEST
16040  101314  000177  177777  177777          .WORD   177,-1, 1,-1            ;AC0
16041  101322  177777
16042  101324  177600                          .WORD   177600          ;RESULT
16043  101326  007700                          .WORD   7700            ; TEST FPS
16044  101330  007710                          .WORD   7710            ;RESULT FPS
16045                                  ;4/EXP=-100
16046  101332  004767  000066                  JSR     R7,SXPSUB               ;DO TEST
16047  101336  020000  000000  177777          .WORD   20000,0,-1,-1           ;AC0
16048  101344  177777
16049  101346  177700                          .WORD   -100            ;RESULT
16050  101350  040200                          .WORD   40200           ; TEST FPS
16051  101352  040210                          .WORD   40210           ;RESULT FPS
16052                                  ;5/EXP=200
16053  101354  004767  000044                  JSR     R7,SXPSUB               ;DO TEST
16054  101360  040000  000000  000000          .WORD   40000,0,0,0             ;AC0
16055  101366  000000
16056  101370  000000                          .WORD   0               ;RESULT
16057  101372  007700                          .WORD   7700            ; TEST FPS
16058  101374  007704                          .WORD   7704    ;RESULT FPS
16059                                  ;6/EXP=0
16060  101376  004767  000022                  JSR     R7,SXPSUB               ;DO TEST
16061  101402  000177  177777  177777          .WORD   177,-1,-1,-1            ;AC0
16062  101410  177777
16063  101412  177600                          .WORD   177600          ;RESULT
16064  101414  000000                          .WORD   0               ; TEST FPS
16065  101416  000010                          .WORD   10              ;RESULT FPS
16066                                  ;
16067
16068  101420  000167  000104                  JMP     HOP22           ;GET OVER SUBROUTINE
16069                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16070                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16071                                  ;STEXP
16072                                  ;               AC0
16073                                  ;               EXPONENT RESULT
16074                                  ;               FPS BEFORE EXECUTION
16075                                  ;               FPS AFTER EXECUTION
16076                                  ;
16077                                  ;NO TRAPS CAN OCCUR
16078                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16079                                  ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16080                                  ;
16081  101424  012602                  SXPSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
16082  101426  012737  101516  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
16083  101434  012701  003102                  MOV     #RECDST,R1              ;POINT TO RESULT AREA
16084  101440  012700  000200                  MOV     #200,R0                 ;SET FPS TO DOUBLE
16085  101444  170100                          LDFPS   R0                      ;
16086  101446  010204                          MOV     R2,R4                   ;POINT TO AC0 DATA
16087  101450  172414                          LDD     (R4),AC0                        ;LOAD AC0
```

```
16088  101452  016200  000012                MOV     12(R2),R0               ;GET TEST FPS
16089  101456  170100                        LDFPS   R0                      ;LOAD TEST FPS
16090                                        ;
16091  101460  175011                        40$:    STEXP   AC0,(R1)        ;*TEST INSTRUCTION(ACCORDING TO MODE)
16092                                        ;
16093                                        ;VERIFY STATUS
16094  101462  170203                        2$:     STFPS   R3              ;SAVE FPS
16095  101464  016200  000014                        MOV     14(R2),R0               ;GET EXPECTED STATUS
16096  101470  020003                                CMP     R0,R3           ;VERIFY STATUS
16097  101472  001401                                BEQ     3$              ;BRANCH IF GOOD
16098  101474  104003                                ERROR   .3              ;FPP ERROR
16099                                                                          ;BAD FPS
16100  101476  016204  000010                3$:     MOV     10(R2),R4               ;POINT TO EXPECTED EXPONENT
16101  101502  020437  003102                        CMP     R4,@#RECDST             ;VERIFY EXPONENT
16102  101506  001401                                BEQ     5$              ;BRANCH IF GOOD
16103  101510  104003                                ERROR   .3              ;FPP ERROR
16104                                                                          ;BAD AC0
16105  101512  000162  000016                5$:     JMP     16(R2)          ;RETURN FROM TEST
16106                                        ;
16107                                        ;INSTRUCTION TRAPPED
16108  101516  012600                        50$:    MOV     (SP)+,R0        ;SAVE PC
16109  101520  012605                                MOV     (SP)+,R5        ;SAVE OLD PS
16110  101522  104003                                ERROR   .3              ;FPP ERROR
16111                                                                          ;WILD TRAP DURING STEXP
16112  101524  000167  177762                        JMP     5$              ;REJOIN CODE
16113                                        ;
16114
16115  101530                                HOP22:
16116
16117
16118                                        .ENABL AMA
16119                                        .SBTTL  TEST  - CHECK REGISTER ACCESS
16120                                        ;CHECK REGISTER ACCESS - THIS TEST WILL VERIFY EACH OF THE THREE
16121                                        ;CACHE MEMORY SYSTEM REGISTERS CAN BE ACCESSED WITHOUT A TRAP TO
16122                                        ;LOCATION 4(NON-EXISTANT ADDRESS TRAP) OCCURRING. THE REGISTERS TO
16123                                        ;BE TESTED ARE:
16124                                        ;       CACHE CONTROL           17777746
16125                                        ;       MEMORY SYSTEM ERROR     17777744
16126                                        ;       HIT/MISS                17777752
16127                                        ;EACH REGISTER WILL BE ACCESSED WITH A WRITE CYCLE.
16128                                        ;
16129                                        ;BGNTST
16130                                        ;SAVE CONTENT OF LOCATION 4
16131                                        ;LET LOCATION 4 POINT TO ERROR ROUTINE
16132                                        ;INITIALIZE R TO TOP OF ADDRESS TABLE
16133                                        ;DO UNTIL FOR ALL REGISTERS
16134                                        ;.      TEST REGISTER UNDER TEST
16135                                        ;.      IF TIMEOUT DID HAPPEN
16136                                        ;.              ERROR
16137                                        ;.      ENDIF
16138                                        ;ENDDO
16139                                        ;RESTORE CONTENTS OF LOCATION 4
16140                                        ;EXIT   TST
16141                                        ;
16142                                        ;ADDRESS TABLE: 17777746
16143                                        ;               17777744
```

```
16144                                 ;               17777752
16145                                 ;
16146                                 ;ENDTST
16147                                 ;ERROR ROUTINE: SET TIMEOUT FLAG
16148                                 ;               RETURN
16149
16150                                 ;;*****************************************************************
16151  101530  000004                 TST2:   SCOPE
16152  101532  012737  132334  000004         MOV     #TOUT,  @#4
16153  101540  013737  000004  002762         MOV     @#4,    SLOC00          ;SAVE CONTENTS OF VECTOR 4
16154  101546  012737  101652  000004         MOV     #ERROUT,@#4             ;LET VECTOR 4 POINT TO ERROR ROUTINE
16155  101554  005001                         CLR     R1                      ;CLEAR TIMEOUT FLAG
16156  101556  005737  177746         1$:     TST     CCR                     ;READ REGISTER UNDER TEST
16157  101562  005701                         TST     R1                      ;TIMEOUT?
16158  101564  001404                         BEQ     2$
16159  101566  012737  177746  001122         MOV     #CCR,$BDADR             ;STORE LOCATION
16160  101574  104130                         ERROR   +130                    ;TIMEOUT ACCESING CCR
16161  101576  005001                 2$:     CLR     R1                      ;CLEAR TIMOUT FLAG
16162  101600  005737  177744                 TST     MSER                    ;READ MSER
16163  101604  005701                         TST     R1                      ;TIMEOUT ?
16164  101606  001404                         BEQ     3$
16165  101610  012737  177744  001122         MOV     #MSER,$BDADR            ;STORE LOCATION
16166  101616  104130                         ERROR   +130                    ;TIMEOUT ACCESSING MSER
16167  101620  005001                 3$:     CLR     R1                      ;CLEAR TIMEOUT FLAG
16168  101622  005737  177752                 TST     HITMIS                  ;ACCESS HIT/MISS
16169  101626  005701                         TST     R1                      ;TIMEOUT?
16170  101630  001404                         BEQ     4$
16171  101632  012737  177752  001122         MOV     #HITMIS,$BDADR          ;STORE LOCATION
16172  101640  104130                         ERROR   +130                    ;TIMEOUT ACCESSING HIT/MISS
16173  101642  013737  002762  000004 4$:     MOV     SLOC00, @#4             ;RESTORE VECTOR 4
16174  101650  000402                         BR      TST3            ;;GO TO NEXT TEST
16175
16176  101652                         ERROUT:
16177  101652  005201                         INC     R1                              ;FLAG TIMEOUT
16178  101654  000002                         RTI
16179
16180
```

```
                                    .SBTTL  TEST  - CCR REGISTER BIT TEST
                                    ;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 BIT 15-11 OR BIT 8 THEN
                                    ;.      .       .       .       ERROR IN READ/WRITE BITS OF CCR
                                    ;.      .       .       ENDIF
                                    ;.      ENDIF
                                    ;.      UPDATE TO NEXT BIT
                                    ;ENDDO
                                    ;ENDTST

                                    ;;*****************************************************************
101656  000004                      TST3:   SCOPE
101660  012701  000001                      MOV     #1,     R1              ;INITIALIZE GOOD DATA TO #1
101664  005037  177746                      CLR     CCR                     ;CLEAR CCR
101670  010137  177746              1$:     MOV     R1,     CCR             ;WRITE GOOD DATA TO CCR
101674  023701  177746                      CMP     CCR,    R1              ;IF CCR NOT EQUAL GOOD DATA
101700  001407                              BEQ     3$                      ;THEN
101702  005737  177746                      TST     CCR                     ;IF CCR EQUAL TO ZERO
101706  001003                              BNE     2$                      ;THEN
101710  032701  174400                      BIT     #174400,R1              ;IF GOOD DATA NE TO BIT 15-11 OR BIT 8
101714  001001                              BNE     3$                      ;THEN
101716  104004                      2$:     ERROR   .4                      ;ERROR IN READ/WRITE BITS OF CCR
101720  006101                      3$:     ROL     R1                      ;UPDATE TO NEXT BIT
101722  103362                              BCC     1$                      ;DO UNTIL ALL BITS TESTED
```

```
16218                                   .SBTTL  TEST  - FORCE MISS TEST
16219                                   ;FORCE MISS TEST - THIS TEST WILL VERIFY THAT ALL REFERENCES MADE
16220                                   ;WITH EITHER BIT<3> OR BIT<2> OF THE CCR SET CAUSE A CACHE MISS AND
16221                                   ;LEAVE THE CACHE ENTRY UNCHANGED. FIRST WRITE A TEST ADDRESS WITH
16222                                   ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN
16223                                   ;INTO THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH
16224                                   ;NEW DATA. NEXT CLEAR BITS<3:2> AND READ THE TEST ADDRESS, THE DATA
16225                                   ;SHOULD EQUAL TO PATTERN 1. FINALLY READ THE TEST ADDRESS WITH BIT<3>
16226                                   ;SET, THE DATA SHOULD EQUAL PATTERN 2. A LAST WRITE MUST BE DONE WITH
16227                                   ;BOTH FORCE BITS CLEARED BECAUSE THE CACHE AND MAIN MEMORY HAVE
16228                                   ;DIFFERENT DATA.
16229                                   ;
16230                                   ;
16231                                   ;BGNTST
16232                                   ;WRITE TEST ADDRESS WITH PATTERN 1
16233                                   ;SET CCR BITS<3:2> = 0,1
16234                                   ;WRITE TEST ADDRESS WITH PATTERN 2
16235                                   ;CLEAR CCR BIT<3>
16236                                   ;READ TEST ADDRESS
16237                                   ;SAVE HIT/MISS REGISTER DATA
16238                                   ;COMPARE RECEIVED DATA TO PATTERN 1
16239                                   ;IF DATA NOT EQUAL THEN
16240                                   ;.      IF DATA EQUAL TO PATTERN 2 THEN
16241                                   ;.      .       IF HIT THEN
16242                                   ;.      .       .       ERROR FORCE MISS WRITES TO CACHE
16243                                   ;.      .       ELSE
16244                                   ;.      .       .       IF PARITY ERROR INDICATORS EQUAL 0 THEN
16245                                   ;.      .       .       .       ERROR FROCE MISS INVALIDATES CACHE
16246                                   ;.      .       .       ELSE
16247                                   ;.      .       .       .       IF ALL PARITY INDICATORS SET THEN
16248                                   ;.      .       .       .       .       ERROR PARITY ABORT AFTER FORCE MISS
16249                                   ;.      .       .       .       ENDIF
16250                                   ;.      .       .       .       IF TAG PARITY ERROR THEN
16251                                   ;.      .       .       .       .       ERROR  TAG PARITY ERROR AFTER FORCE MISS
16252                                   ;.      .       .       .       ELSE
16253                                   ;.      .       .       .       .       IF B0 AND  B1 ERROR THEN
16254                                   ;.      .       .       .       .       .       ERROR DATA PARITY ERROR AFTER FORCE MISS
16255                                   ;.      .       .       .       .       ENDIF
16256                                   ;.      .       .       .       .       IF B0 PARITY ERROR THEN
16257                                   ;.      .       .       .       .       .       ERROR LOW BYTE PARITY ERROR
16258                                   ;.      .       .       .       .       ENDIF
16259                                   ;.      .       .       .       .       IF B1 PARITY ERROR THEN
16260                                   ;.      .       .       .       .       .       ERROR HIGH BYTE PARITY ERROR
16261                                   ;.      .       .       .       .       ENDIF
16262                                   ;.      .       .       .       ENDIF
16263                                   ;.      .       .       ENDIF
16264                                   ;.      .       ENDIF
16265                                   ;.      ELSE
16266                                   ;.      .       ERROR IN DATA PATH
16267                                   ;.      ENDIF
16268                                   ;ENDIF
16269                                   ;SET CCR<3:2> = 1,0
16270                                   ;READ TEST ADDRESS
16271                                   ;SAVE HIT/MISS REGISTER DATA
16272                                   ;COMPARE RECEIVED DATA TO PATTERN 2
16273                                   ;IF DATA NOT EQUAL THEN
```

```
16274                                          ;.       IF RECIEVED DATA EQUAL TO PATTERN 1 THEN
16275                                          ;.       .       IF HIT THEN
16276                                          ;.       .       .       ERROR FORCE MISS READS FROM CACHE
16277                                          ;.       .       ELSE
16278                                          ;.       .       .       ERROR FORCE MISS READS FROM CACHE AND MISS
16279                                          ;.       .       ENDIF
16280                                          ;.       ELSE
16281                                          ;.       .       ERROR IN DATA PATH
16282                                          ;.       ENDIF
16283                                          ;ENDIF
16284                                          ;CLEAR CCR<3:2>
16285                                          ;WRITE TEST ADDRESS
16286                                          ;ENDTST
16287                                          ;
16288                                          ;;*****************************************************************
16289  101724  000004                        TST4:   SCOPE
16290  101726  013737  000114  002762                MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16291  101734  012737  102230  000114                MOV     #FMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
16292  101742  005003                                CLR     R3                      ;CLEAR MSER SAVE LOCATION
16293  101744  005037  003122                        CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16294  101750  012737  000004  177746                MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
16295  101756  012737  177777  003122                MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16296  101764  012737  000200  177746                MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16297  101772  012737  177777  001124                MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
16298  102000  013701  003122                        MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
16299  102004  013702  177752                        MOV     @#HITMIS,R2             ;SAVE HIT/MISS DATA
16300  102010  005701                                TST     R1                      ;COMPARE RECEIVED DATA TO PATTERN 1
16301  102012  001451                                BEQ     1$                      ;IF DATAS NOT EQUAL THEN
16302  102014  022701  177777                        CMP     #177777,R1              ;IF DATA EQUAL TO PATTERN 2
16303  102020  001045                                BNE     106$                    ;THEN
16304  102022  032702  000002                        BIT     #BIT01, R2              ;IF HIT
16305  102026  001402                                BEQ     100$                    ;THEN
16306  102030  104005                                ERROR   .5                      ;FORCE MISS WRITES TO CACHE
16307  102032  000441                                BR      1$                      ;ELSE
16308  102034  032703  100340          100$:   BIT     #100340,R3              ;IF PARITY INDICATORS EQUAL 0
16309  102040  001002                                BNE     101$                    ;THEN
16310  102042  104006                                ERROR   .6                      ;FORCE MISS WRITE INVALIDATES CACHE
16311  102044  000434                                BR      1$                      ;ELSE
16312  102046  022703  100340          101$:   CMP     #100340,R3              ;IF ALL PARITY INDICATORS SET
16313  102052  001002                                BNE     102$                    ;THEN
16314  102054  104007                                ERROR   .7                      ;PARITY ABORT AFTER FORCE MISS
16315  102056  000427                                BR      1$                      ;ELSE
16316  102060  032703  000040          102$:   BIT     #BIT05, R3              ;IF TAG PARITY ERROR
16317  102064  001402                                BEQ     103$                    ;THEN
16318  102066  104010                                ERROR   .10                     ;TAG PARITY ERROR AFTER FORCE MISS
16319  102070  000422                                BR      1$                      ;ELSE
16320  102072  010304          103$:   MOV     R3,     R4              ;COPY MSER INTO REG 4
16321  102074  042704  177477                        BIC     #177477,R4              ;MASK FOR B0 AND B1 DATA
16322  102100  022704  000300                        CMP     #300,   R4              ;IF B0 AND B1 DATA
16323  102104  001002                                BNE     104$                    ;THEN
16324  102106  104011                                ERROR   .11                     ;DATA PARITY ERROR AFTER FORCE MISS
16325  102110  000412                                BR      1$                      ;ELSE
16326  102112  032703  000100          104$:   BIT     #100,   R3              ;IF B0 ERROR
16327  102116  001401                                BEQ     105$                    ;THEN
16328  102120  104012                                ERROR   .12                     ;LOW BYTE PARITY ERROR AFTER FORCE MISS
16329  102122  032703  000200          105$:   BIT     #200,   R3              ;IF B1 ERROR
```

```
 16330  102126  001403                          BEQ     1$                      ;THEN
 16331  102130  104013                          ERROR   +13                     ;HIGH BYTE PARITY ERROR AFTER FORCE MISS
 16332  102132  000401                          BR      1$                      ;ENDIF
 16333  102134  104014                  106$:   ERROR   +14                     ;ERROR DATA PATH
 16334  102136  005003                  1$:     CLR     R3                      ;CLEAR MSER SAVE REGISTER
 16335  102140  052737  000010  177746          BIS     #BIT03, CCR             ;SET CCR BITS <3:2> = 1,0
 16336  102146  013701  003122                  MOV     @#TSTLOC,R1             ;READ TEST LOCATION
 16337  102152  013702  177752                  MOV     @#HITMIS,R2             ;SAVE HIT/MISS REGISTER DATA
 16338  102156  020127  177777                  CMP     R1,     #177777         ;COMPARE RECEIVED DATA TO PATTERN 2
 16339  102162  001412                          BEQ     2$                      ;IF DATAS NOT EQUAL THEN
 16340  102164  005701                          TST     R1                      ;IF RECIEVED DATA EQUAL TO PATTERN 1
 16341  102166  001007                          BNE     108$                    ;THEN
 16342  102170  032702  000002                  BIT     #BIT01, R2              ;IF HIT
 16343  102174  001402                          BEQ     107$                    ;THEN
 16344  102176  104015                          ERROR   +15                     ;FORCE MISS READS FROM CACHE
 16345  102200  000403                          BR      2$                      ;ELSE
 16346  102202  104016                  107$:   ERROR   +16                     ;FORCE MISS READS FROM CACHE AND MISS
 16347  102204  000401                          BR      2$                      ;ENDIF
 16348  102206  104014                  108$:   ERROR   +14                     ;ERROR IN DATA PATH
 16349  102210  013737  002762  000114  2$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
 16350  102216  005037  177746                  CLR     CCR                     ;CLEAR CCR BITS<3:2>
 16351  102222  005037  003122                  CLR     @#TSTLOC                ;WRITE TEST ADDRESS
 16352  102226  000405                          BR      TST5            ;;GO TO NEXT TEST
 16353
 16354
 16355  102230  011637  001122          FMPARR: MOV     (SP),$BDADR             ;SAVE ADDRESS THAT CAUSED ABORT
 16356  102234  013703  177744                  MOV     MSER,   R3              ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
 16357  102240  000002                          RTI                             ;REGISTER AND RETURN
 16358
```

```
 16359                                  .SBTTL  T ST  - HIT/MISS REGISTER TEST PART 1
 16360                                  ;HIT/MIS  REGISTER TEST PART 1 - THIS TEST WILL VERIFY THAT THE HIT/MISS
 16361                                  ;REGISTE  CORRECTLY LOGS HITS AND MISSES.FIRST WRITE A TEST ADDRESS WITH
 16362                                  ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN INTO
 16363                                  ;THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH NEW DATA.
 16364                                  ;NEXT CLEAR CCR BITS<3:2> AND READ THE TEST ADDRESS. THE HIT/MISS REGISTER
 16365                                  ;SHOULD LOGGED A HIT. FINALLY READ THE TEST ADDRESS PLUS 20000(8) THE
 16366                                  ;HIT/MISS REGISTER SHOULD HAVE LOGGED A MISS.
 16367                                  ;
 16368                                  ;
 16369                                  ;BGNTST
 16370                                  ;WRITE TEST ADDRESS WITH PATTERN 1
 16371                                  ;SET CCR BITS<3:2> = 0,1
 16372                                  ;WRITE TEST ADDRESS WITH PATTERN 2
 16373                                  ;CLEAR CCR BIT<3>
 16374                                  ;READ TEST ADDRESS
 16375                                  ;IF HIT/MISS REGISTER BIT 1 NOT SET THEN
 16376                                  ;.      ERROR IN RECORDING HITS IN HIT/MISS
 16377                                  ;ENDIF
 16378                                  ;READ TEST ADDRESS + 20000(8)
 16379                                  ;IF HIT/MISS REGISTER BIT 1 SET THEN
 16380                                  ;.      ERROR IN RECORDING HITS IN HIT/MISS
 16381                                  ;ENDIF
 16382                                  ;ENDTST
 16383                                  ;
 16384                                  ;;*****************************************************************
 16385  102242  000004                 TST5:   SCOPE
 16386  102244  013737  000114  002762         MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
 16387  102252  012737  102366  000114         MOV     #HMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
 16388  102260  005037  003122                 CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
 16389  102264  012737  000004  177746         MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
 16390  102272  012737  177777  003122         MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
 16391  102300  012737  000200  177746         MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
 16392  102306  013701  003122                 MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
 16393  102312  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE REGISTER
 16394  102320  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 NOT SET
 16395  102326  001001                         BNE     1$                      ;THEN
 16396  102330  104045                         ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 16397  102332  013701  023122          1$:    MOV     @#TSTLOC+8192.,R1       ;READ TEST LOCATION + 20000(8)
 16398  102336  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE REGISTER
 16399  102344  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
 16400  102352  001401                         BEQ     2$                      ;THEN
 16401  102354  104045                         ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 16402  102356  013737  002762  000114  2$:    MOV     SLOC00, @#114           ;RESTORE VECTOR 114
 16403  102364  000403                         BR      TST6            ;;GO TO NEXT TEST
 16404
 16405  102366  013703  177744          HMPARR: MOV     MSER,   R3              ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
 16406  102372  000002                         RTI                             ;REGISTER AND RETURN
 16407
```

```
                                        .SBTTL  TEST  - HIT/MISS REGISTER TEST
                                        ;HIT/MISS REGISTER TEST - THIS TEST WILL VERIFY THAT THE HIT/MISS
                                        ;REGISTER CORRECTLY LOGS CACHE HITS AND MISSES. IT WILL ALSO VERIFY
                                        ;THAT EACH BIT OF THE REGISTER IS UNIQUE. THIS WILL BE DONE BY
                                        ;FLOATING A ZERO THROUGH A FIELD OF ONES. THE ROTATING WILL BE DONE
                                        ;BY EXECUTING A TST @#HM (WHERE HM IS THE ADDRESS OF THE HIT/MISS
                                        ;REGISTER) AT SUCCESSIVE POSITIONS IN A SET OF EIGHT NOPS. THE READ
                                        ;OF THE I/O PAGE ADDRESS OF THE HIT/MISS REGISTER SHOULD CAUSE A
                                        ;MISS TO BE RECORDED.
                                        ;
                                        ;BGNTST
                                        ;INITIALIZE LOOP INDICATOR
                                        ;INITIALIZE EXPECTED DATA
                                        ;DO UNTIL LOOP INDICATOR = SEVEN
                                        ;.      PUT BACKGROUND DATA INTO EXECUTION BUFFER
                                        ;.      PUT TST INSTRUCTION INTO TEST LOCATION
                                        ;.      JUMP TO EXECUTE BUFFER
                                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                                        ;.              ERROR IN RECORDING HITS IN HIT/MISS
                                        ;.      ENDIF
                                        ;.      INCREMENT LOOP INDICATOR
                                        ;.      UPDATE EXPECTED DATA
                                        ;ENDDO
                                        ;EXIT   TST
                                        ;
                                        ;BACKGROUND DATA:NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               NOP
                                        ;               MOV     @#HM,R2
                                        ;               RTS     PC
                                        ;ENDTST
                                        ;;*****************************************************************
102374  000004                  TST6:   SCOPE
102376  005037  003114                  CLR     LOOPIN                  ;INITIALIZE LOOP INDICATOR
102402  013737  023144  001162          MOV     @#TSTLOC+20022,$TMP1    ;SAVE FOR LATER
102410  012703  003124                  MOV     #TSTLOC+2,R3            ;GET ADDRESS OF EXECUTION BUFFER
102414  012704  102564                  MOV     #EXPTBL,R4              ;GET ADDRESS OF EXPECTED DATA TABLE
102420  012705  177752                  MOV     #HITMIS,R5              ;PUT ADDRESS OF HIT/MISS REGISTER IN GPR
102424  023727  003114  000007  1$:     CMP     LOOPIN, #7              ;DO UNTIL LOOP INDICATOR EQUALS 7
102432  001440                          BEQ     ENDMRT                  ;THEN EXIT
102434  012702  000013                  MOV     #13,    R2
102440  012700  102536                  MOV     #BACDAT,R0              ;PUT BACKGROUND DATA INTO
102444  012701  003122                  MOV     #TSTLOC,R1              ;EXECUTION BUFFER
102450  012021                  2$:     MOV     (R0)+,  (R1)+
102452  077202                          SOB     R2,     2$              ;LOOP FOR TEN WORDS
102454  023727  003114  000006          CMP     LOOPIN, #6              ;IS THIS LAST LOOP
102462  001004                          BNE     3$                      ;IF YES THEN
102464  013737  001162  023144          MOV     $TMP1,@#TSTLOC+20022    ;INVALIDATE FETCH OF "RECDAT"
102472  000404                          BR      4$                      ;ELSE
102474  012723  005737          3$:     MOV     #5737,  (R3)+           ;MOVE "TST" INSTRUCTION TO BUFFER
102500  012713  177752                  MOV     #HITMIS,(R3)            ;MOVE HIT.MISS REG. ADD. TO BUFFER
```

```
16464   102504   004737   003122          4$:     JSR     PC,     TSTLOC          ;GO EXECUTE TEST CODE
16465   102510   021437   110152                  CMP     (R4),   RECDAT         ;IF EXPECTED DATA NOT EQUAL TO
16466   102514   001403                           BEQ     5$                      ;RECEIVED DATA THEN
16467   102516   011437   001124                  MOV     (R4),$GUDAT             ;SAVE EXPECTED PATTERN
16468   102522   104017                           ERROR   .17                     ;ERROR IN RECORDING HITS IN HIT/MISS
16469   102524   005724                  5$:     TST     (R4)+                   ;INCREMENT POINTER TO EXPECTED PATTERN
16470   102526   005237   003114                  INC     LOOPIN                  ;INCREMENT LOOP COUNTER
16471   102532   000734                           BR      1$
16472   102534                           ENDHRT:
16473   102534   000422                           BR      TST7            ;;GO TO NEXT TEST
16474
16475   102536   000240                  BACDAT: .WORD   NOP
16476   102540   000240                          .WORD   NOP
16477   102542   000240                          .WORD   NOP
16478   102544   000240                          .WORD   NOP
16479   102546   000240                          .WORD   NOP
16480   102550   000240                          .WORD   NOP
16481   102552   000240                          .WORD   NOP
16482   102554   000240                          .WORD   NOP
16483   102556   011537   110152                 MOV     (R5),RECDAT             ;SAVE CONTENTS OF HIT/MISS REGISTER
16484   102562   000207                          RTS     PC
16485
16486   102564   000077                  EXPTBL: .WORD   77
16487   102566   000037                          .WORD   37
16488   102570   000057                          .WORD   57
16489   102572   000067                          .WORD   67
16490   102574   000073                          .WORD   73
16491   102576   000075                          .WORD   75
16492   102600   000076                          .WORD   76
16493
```

```
16494                                   .SBTTL  TEST  - BYTE ALLOCATION TEST
16495                                   ;BYTE ALLOCATION TEST - THIS TEST WILL VERIFY THAT THE CACHE SYSTEM CORRECTLY
16496                                   ;HANDLES BYTE ACCESSES. THE TEST WILL FIRST CHECK THAT A WRITE ACCESS WHICH
16497                                   ;IS A MISS DOES NOT UPDATE THE CACHE. THIS WILL BE DONE BY FIRST ASSURRING
16498                                   ;A MISS AT A TEST LOCATION. THEN A WRITE BYTE ACCESS WILL BE DONE. FINALLY
16499                                   ;THE LOCATION WILL BE READ. THE WRITE BYTE SHOULD NOT HAVE ALLOCATED THE
16500                                   ;ADDRESS. NEXT THE TEST WILL VERIFY THAT IF A WRITE BYTE ACCESS IS A HIT
16501                                   ;THAT THE CACHE LOCATION IS UPDATED. THE TEST WILL FIRST WRITE A TEST LOCATION
16502                                   ;WITH A PATTERN TO ALLOCATE THE ADDRESS. THEN A SECOND PATTERN WILL BE WRITTEN
16503                                   ;TO THE HIGH BYTE OF THE TEST LOCATION. THE TEST LOCATION WILL THEN BE READ.
16504                                   ;IF THE HIGH BYTE OF THE TEST LOCATION IS EQUAL TO THE SECOND PATTERN THEN THE
16505                                   ;LOCATION WAS UPDATED. THE SECOND TEST WILL BE REPEATED TO TEST LOW BYTE
16506                                   ;ACCESSES.
16507                                   ;
16508                                   ;BGNTST
16509                                   ;SAVE VECTOR 114
16510                                   ;LET VECTOR 114 POINT TO BYTE PARITY ROUTINE
16511                                   ;SET PARITY ABORT BIT IN CACHE CONTROL REGISTER
16512                                   ;READ TEST LOCATION + 4K
16513                                   ;WRITE LOW BYTE TO TEST ADDRESS
16514                                   ;READ LOW BYTE OF TEST ADDRESS
16515                                   ;IF HIT/MISS REGISTER BIT 1 SET THEN
16516                                   ;.      ERROR WRITE BYTE ALLOCATES THE CACHE
16517                                   ;ENDIF
16518                                   ;WRITE PATTERN 1 TO TEST LOCATION
16519                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION+1
16520                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16521                                   ;.      IF B0 PARITY ERROR THEN
16522                                   ;.      .       ERROR LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16523                                   ;.      ELSE
16524                                   ;.      .       IF B1 PARITY ERROR THEN
16525                                   ;.      .       .       ERROR HIGH BYTE PARITY ERROR ON WRITE BYTE HIT
16526                                   ;.      .       ELSE
16527                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16528                                   ;.      .       ENDIF
16529                                   ;.      ENDIF
16530                                   ;ELSE
16531                                   ;.      READ TEST LOCATION
16532                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16533                                   ;.      .       IF BYTE DATA REVERSED THEN
16534                                   ;.      .       .       ERROR BYTES REVERSED ON WRITE CYCLES
16535                                   ;.      .       ELSE
16536                                   ;.      .       .       IF HIGH BYTE DATA ZERO THEN
16537                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16538                                   ;.      .       .       ELSE
16539                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16540                                   ;.      .       .       ENDIF
16541                                   ;.      .       ENDIF
16542                                   ;.      ENDIF
16543                                   ;ENDIF
16544                                   ;WRITE PATTERN 1 TO TEST LOCATION
16545                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION LOW BYTE
16546                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16547                                   ;.      IF B0 PARITY ERROR THEN
16548                                   ;.      .       ERROR LOW BYTE PARITY
16549                                   ;.      ELSE
```

```
16550                                   ;.      .       IF B1 PARITY ERROR THEN
16551                                   ;.      .       .       ERROR HIGH BYTE PRITY
16552                                   ;.      .       ELSE
16553                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16554                                   ;.      .       ENDIF
16555                                   ;.      ENDIF
16556                                   ;ELSE
16557                                   ;.      READ TEST LOCATION
16558                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16559                                   ;.      .       IF BYTE DATA REVERSED THEN
16560                                   ;.      .       .       ERROR BYTES REVERSED
16561                                   ;.      .       ELSE
16562                                   ;.      .       .       IF LOW BYTE DATA ZERO THEN
16563                                   ;.      .       .       .       ERROR IN DATA PATH
16564                                   ;.      .       .       ELSE
16565                                   ;.      .       .       .       ERROR IN DATA PATH
16566                                   ;.      .       .       ENDIF
16567                                   ;.      .       ENDIF
16568                                   ;.      ENDIF
16569                                   ;ENDIF
16570                                   ;CLEAR CACHE CONTROL REGISTER
16571                                   ;RESTORE VECTOR 114
16572                                   ;EXIT   TST
16573                                   ;
16574                                   ;BYTE PARITY ROUTINE:   SAVE MSER
16575                                   ;                       RETURN
16576                                   ;ENDTST
16577
16578                                   ;;*****************************************************************
16579  102602  000004                   TST7:   SCOPE
16580  102604  013737  000114  002762           MOV     @#114,  SLOC00          ;SAVE VECTOR 114
16581  102612  012737  103012  000114           MOV     #BYPARR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
16582  102620  005003                           CLR     R3                      ;CLEAR MSER SAVE REGISTER
16583  102622  012737  000200  177746           MOV     #BIT07, CCR             ;SET PARITY ABORT BIT IN CCR
16584  102630  005737  023122                   TST     TSTLOC+8192.            ;READ TEST LOCATION + 4K TO CAUSE MISS
16585  102634  105037  003122                   CLRB    TSTLOC                  ;WRITE LOW BYTE OF TEST LOCATION
16586  102640  105737  003122                   TSTB    TSTLOC                  ;READ LOW BYTE OF TEST LOCATION
16587  102644  013737  177752  110152           MOV     HITMIS,RECDAT           ;STORE REGISTER
16588  102652  032737  000004  110152           BIT     #BIT02,RECDAT           ;IF BIT1 OF HIT/MISS REGISTER SET
16589  102660  001401                           BEQ     1$                      ;THEN
16590  102662  104020                           ERROR   +20                     ;WRITE BYTE ALLOCATES CACHE
16591  102664  005037  003122           1$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16592  102670  152737  000377  003123           BISB    #377,   @#TSTLOC+1      ;WRITE PATTERN 2 TO HIGH BYTE
16593  102676  013737  177752  110152           MOV     HITMIS,RECDAT           ;STORE REGISTER
16594  102704  032737  000004  110152           BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16595  102712  001002                           BNE     2$                      ;THEN
16596  102714  104021                   101$:   ERROR   +21                     ;WRITE BYTE HIT DOES NOT RECORD HIT
16597  102716  000405                           BR      3$                      ;ELSE
16598  102720  022737  177400  003122   2$:     CMP     #177400,@#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
16599  102726  001401                           BEQ     3$                      ;PATTERN 2 THEN
16600  102730  104022                           ERROR   +22                     ;BYTES REVERSED ON WRITE CYCLES
16601  102732  005037  003122           3$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16602  102736  152737  000377  003122           BISB    #377,   @#TSTLOC        ;WRITE PATTERN 2 TO LOW BYTE
16603  102744  013737  177752  110152           MOV     HITMIS,RECDAT           ;STORE REGISTER
16604  102752  032737  000004  110152           BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16605  102760  001001                           BNE     4$                      ;THEN
```

```
 16606  102762  104012                          ERROR   ·12                      ;LOW BYTE PARITY ERROR ON WRITE BYTE HIT
 16607  102764  022737  000377  003122  4$:     CMP     @#377,   @#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
 16608  102772  001401                          BEQ     5$                       ;PATTERN 2 THEN
 16609  102774  104022                          ERROR   ·22                      ;BYTES REVERSED ON WRITE CYCLES
 16610  102776  005037  177746          5$:     CLR     CCR                      ;CLEAR CCR BEFORE EXIT
 16611  103002  013737  002762  000114          MOV     SLOCOO, @#114            ;RESTORE VECTOR 114
 16612  103010  000405                          BR      TST10           ;;GO TO NEXT TEST
 16613
 16614
 16615  103012  011637  001122          BYPARR: MOV     (SP),    $BDADR          ;SAVE ADDRESS THAT CAUSE ABORT
 16616  103016  013703  177744                  MOV     MSER,    R3              ;SAVE CONTENTS OF MSER
 16617  103022  000002                          RTI                              ;RETURN
 16618
```

```
 16619                                  .SBTTL  TEST  - PDR BIT15 (BYPASS) TEST
 16620                                  ;PDR BIT15 (BYPASS) TEST - THIS TEST WILL VERIFY THAT WHEN BIT<15> IS SET
 16621                                  ;IN A PDR AND AN ACCESS IS MADE TO A LOCATION MAPPED BY THE SELECTED PDR
 16622                                  ;THAT WOULD NORMALLY CAUSE A CACHE ACCESS, THE CACHE IS BYPASSED (I.E. THE
 16623                                  ;CACHE LOCATION IS INVALIDATED AND A MAIN MEMORY IS READ. THIS WILL BE DONE
 16624                                  ;BY FIRST WRITING A TEST LOCATION WITH A KNOWN PATTERN TO ALLOCATE THE ADDRESS.
 16625                                  ;THEN THE LOCATION WILL BE WRITTEN AGAIN WITH THE MMU ENABLED AND BIT <15> SET
 16626                                  ;IN THE CONTROLING PDR WITH A NEW PATTERN. THE LOCATION WILL THEN BE READ
 16627                                  ;AND A MISS SHOULD BE LOGGED IN THE HIT/MISS REGISTER. THIS READ WILL ALSO
 16628                                  ;BRING VALID DATA INTO CACHE FROM MEMORY. THE MMU WILL AGAIN BE ENABLED
 16629                                  ;WITH BIT <15> SET AND A READ CYCLE DONE. THIS SHOULD INVALIDATE THE CACHE.
 16630                                  ;NEXT THE MMU WILL BE DISABLED AND THE LOCATION READ FOR A THIRD TIME. THIS
 16631                                  ;WILL BE DONE WITH BIT<15> STILL SET. A MISS SHOULD ALSO BE LOGGED. LASTLY
 16632                                  ;BIT<15> WILL BE CLEARED AND THE MMU ENABLED AND THE LOCATION AGAIN READ.
 16633                                  ;THIS TIME A CACHE HIT SHOULD BE LOGGED.
 16634                                  ;
 16635                                  ;
 16636                                  ;BGNTST
 16637                                  ;SETUP MMU REGISTERS
 16638                                  ;WRITE TEST LOCATION WITH PATTERN 1
 16639                                  ;SET BIT<15> IN PDR FOR TEST LOCATION
 16640                                  ;ENABLE MMU
 16641                                  ;WRITE TEST LOCATION WITH PATTERN 2
 16642                                  ;DISABLE MMU AND CLEAR BIT<15> IN PDR
 16643                                  ;READ TEST LOCATION
 16644                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
 16645                                  ;.      ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
 16646                                  ;ENDIF
 16647                                  ;SET BIT<15> IN PDR FOR TEST LOCATION
 16648                                  ;ENABLE MMU
 16649                                  ;READ TEST LOCATION (SHOULD INVALIDATE CACHE)
 16650                                  ;DISABLE MMU
 16651                                  ;READ TEST LOCATION
 16652                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
 16653                                  ;.      ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
 16654                                  ;ENDIF
 16655                                  ;CLEAR BIT<15> IN PDR
 16656                                  ;TURN ON MMU
 16657                                  ;READ TEST LOCATION
 16658                                  ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
 16659                                  ;.      ERROR IN RECORDING HITS IN HIT/MISS
 16660                                  ;ENDIF
 16661                                  ;TURN OFF MMU
 16662                                  ;ENDTST
 16663                                  ;
 16664                                  ;;*****************************************************************
 16665  103024  000004                 TST10:  SCOPE
 16666  103026  004737  131362                 JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT REGISTERS
 16667  103032  005037  003122                 CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
 16668  103036  012737  177777  001124         MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
 16669  103044  052737  100000  172300         BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
 16670  103052  005237  177572                 INC     SR0                     ;TURN ON MEMORY MANAGEMENT
 16671  103056  012737  177777  003122         MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
 16672  103064  005037  177572                 CLR     SR0                     ;TURN OFF MEMORY MANAGEMENT
 16673  103070  042737  100000  172300         BIC     #BIT15, KIPDR0          ;CLEAR BIT15 IN PDR FOR TEST LOCATION
 16674  103076  013701  003122                 MOV     @#TSTLOC,R1             ;READ TEST LOCATION
```

```
 16675  103102  013737  177/52  110152          MOV     HITMIS,RECDAT           ;SAVE HIT/MISS REGISTER
 16676  103110  032737  000004  110152          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
 16677  103116  001401                          BEQ     2$                      ;THEN
 16678  103120  104023                          ERROR   .23                     ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
 16679  103122  052737  100000  172300  2$:     BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
 16680  103130  005237  177572                  INC     SR0                     ;TURN ON MEMORY MANAGEMENT
 16681  103134  005737  003122                  TST     TSTLOC                  ;READ TEST LOCATION
 16682  103140  042737  100000  172300          BIC     #BIT15, KIPDR0          ;CLEAR BIT 15 IN PDR
 16683  103146  005037  177572                  CLR     SR0                     ;TURN OFF MMU
 16684  103152  013701  003122                  MOV     @#TSTLOC,R1             ;READ TEST LOCATION
 16685  103156  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16686  103164  032737  000004  110152          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
 16687  103172  001401                          BEQ     3$                      ;THEN
 16688  103174  104023                          ERROR   .23                     ;CONDITIONAL BYPASS DOESN'T INVALIDATE 0 CACHE
 16689  103176  005237  177572          3$:     INC     SR0                     ;TURN ON MEMORY MANAGEMENT
 16690  103202  005737  003122                  TST     @#TSTLOC                ;READ TEST LOCATION ONE MORE TIME
 16691  103206  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16692  103214  032737  000004  110152          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 NOT SET
 16693  103222  001001                          BNE     4$                      ;THEN
 16694  103224  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 16695  103226  042737  000001  177572  4$:     BIC     #BIT00, SR0             ;TURN OFF MMU
 16696
```

```
16697                                   .SBTTL  TEST  - FLUSH CACHE TEST
16698                                   ;FLUSH CACHE TEST - THIS TEST WILL VERIFY THAT WHEN CCR BIT<8> IS
16699                                   ;SET, THE ENTIRE CACHE IS INVALIDATED. FIRST 8K BYTES OF ADDRESSES
16700                                   ;WILL BE READ TO ALLOCATE CACHE. THEN THE CACHE WILL BE FLUSHED.
16701                                   ;THE SAME SET OF ADDRESS WILL THEN BE READ AND THE HIT/MISS REGISTER
16702                                   ;CHECKED AFTER EACH READ FOR A MISS TO BE LOGGED.
16703                                   ;
16704                                   ;BGNTST
16705                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16706                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16707                                   ;DO UNTIL LOOP COUNTER = 0
16708                                   ;.      READ @ADDRESS+
16709                                   ;ENDDO
16710                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16711                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16712                                   ;INITIALIZE MISS COUNT TO 4KWORD COUNT
16713                                   ;FLUSH CACHE
16714                                   ;DO UNTIL LOOP COUNTER = 0
16715                                   ;.      READ @ADDRESS+
16716                                   ;.      ICREMENT ADDRESS TO READ EVERY 2WORDS
16717                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
16718                                   ;.      .       DECREMENT MISS COUNT
16719                                   ;.      ENDIF
16720                                   ;ENDDO
16721                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16722                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16723                                   ;DO UNTIL LOOP COUNTER = 0
16724                                   ;.      READ @ADDRESS+
16725                                   ;ENDDO
16726                                   ;GET LAST ADDRESS OF 8KB BUFFER
16727                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16728                                   ;FLUSH CACHE
16729                                   ;DO UNTIL LOOP COUNTER = 0
16730                                   ;.      READ @ADDRESS-
16731                                   ;.      DECREMENT ADDRESS TO READ EVERY 2WORDS
16732                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
16733                                   ;.      .       DECREMENT MISS COUNT
16734                                   ;.      ENDIF
16735                                   ;ENDDO
16736                                   ;IF MISS COUNT NOT EQUAL TO 4096. THEN
16737                                   ;.      ERROR HITS RECORDED AFTER FLUSHING CACHE
16738                                   ;ENDIF
16739                                   ;ENDTST
16740                                   ;
16741                                   ;;*****************************************************************
16742  103234  000004                  TST11:  SCOPE
16743  103236  004737  131362                  JSR     PC,     INITMM          ;INITIALIZE MMU
16744  103242  012737  001600  172354          MOV     #1600,KIPAR6            ;LET PAR6 POINT TO LOW 28K
16745  103250  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
16746  103254  005237  177572                  INC     SR0                     ;ENABLE MMU
16747  103260  012702  010000                  MOV     #4096.. R2              ;INIT LOOP COUNTER TO 4K WORD COUNT
16748  103264  005721                  1$:     TST     (R1)+                   ;READ ADDRESS TO ALLOCATE CACHE
16749  103266  077202                          SOB     R2,     1$              ;DO UNTIL ALL LOCATIONS ALLOCATED
16750  103270  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
16751  103274  012702  004000                  MOV     #2048.. R2              ;INIT LOOP COUNTER TO 2K WORD COUNT
16752  103300  012703  010000                  MOV     #4096.. R3              ;INITIALIZE MISS COUNT TO 4K
```

```
COKDAA0 KDJ11-B CLUSTER MACY11 30(1046)  23-JAN-84  18:56  PAGE 306
COKDAA.P11    23-JAN-84 18:55           TEST  - FLUSH CACHE TEST                                                    SEQ 0306

 16753  103304  012737  000400  177746          MOV     #400,   CCR             ;FLUSH CACHE
 16754  103312  005721                  2$:     TST     (R1)+                   ;READ ADDRESS
 16755  103314  013737  177752  110152          MOV     @#HITMIS,RECDAT         ;STORE REGISTER
 16756  103322  032737  000004  110152          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER NOT EQUAL
 16757  103330  001401                          BEQ     3$                      ;TO ZERO THEN
 16758  103332  005303                          DEC     R3                      ;DECREMENT MISS COUNT
 16759  103334  062701  000002          3$:     ADD     #2,R1                   ;DO IN 2 WORD INCREMENTS (PMI MEMORY)
 16760  103340  077214                          SOB     R2,     2$              ;LOOP UNTIL ENTIRE CACHE CHECKED
 16761  103342  012702  010000                  MOV     #4096.,R2               ;DO FOR 4K
 16762  103346  012701  140000                  MOV     #140000,R1              ;STARTING WITH 0
 16763  103352  005721                  4$:     TST     (R1)+                   ;ALLOCATE IN CACHE
 16764  103354  077202                          SOB     R2,4$                   ;DO FOR 4K
 16765  103356  012702  004000                  MOV     #2048.,R2               ;DO FOR THE 2ND HALF
 16766  103362  012701  160000                  MOV     #160000,R1              ;START WITH LAST LOCATION
 16767  103366  012737  000400  177746          MOV     #400,CCR                ;FLUSH CACHE
 16768  103374  005741                  5$:     TST     -(R1)                   ;READ ADDRESS
 16769  103376  013737  177752  110152          MOV     @#HITMIS,RECDAT         ;STORE REGISTER
 16770  103404  032737  000004  110152          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER NOT EQUAL
 16771  103412  001401                          BEQ     6$                      ;TO ZERO THEN
 16772  103414  005303                          DEC     R3                      ;DECREMENT MISS COUNT
 16773  103416  162701  000002          6$:     SUB     #2,R1                   ;DO IN 2 WORD DECREMENTS
 16774  103422  077214                          SOB     R2,5$                   ;DO FOR 2K WORDS
 16775  103424  005037  177572                  CLR     SR0                     ;TURN OFF MMU
 16776  103430  022703  010000                  CMP     #4096., R3              ;IF MISS COUNT NOT EQUAL TO 4K WORDS
 16777  103434  001401                          BEQ     8$                      ;THEN
 16778  103436  104024                          ERROR   .24                     ;HITS RECORDED AFTER FLUSHING CACHE
 16779  103440                          8$:
 16780  103440  000400                          BR      TST12           ;;GO TO NEXT TEST
 16781
```

```
 16782                                  .SBTTL  TEST  - UNCONDITIONAL BYPASS TEST
 16783                                  ;UNCONDITIONAL BYPASS TEST - THIS TEST WILL VERIFY THAT WHEN CCR
 16784                                  ;BIT<9> IS SET, A MEMORY REFERENCE IS FORCED TO MAIN MEMORY. THIS
 16785                                  ;WILL ALSO VERIFY THAT READ AND WRITE HIT INVALIDATE THE CACHE
 16786                                  ;LOCATIONS WHEN BYPASS IS SET.
 16787                                  ;
 16788                                  ;BGNTST
 16789                                  ;READ TEST LOCATIONS TO SETUP POSSIBILITY OF HIT
 16790                                  ;SET CCR BIT<9>
 16791                                  ;READ FIRST TEST LOCATION
 16792                                  ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
 16793                                  ;.      ERROR IN RECORDING HITS IN HIT/MISS
 16794                                  ;ENDIF
 16795                                  ;WRITE SECOND TEST LOCATION
 16796                                  ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
 16797                                  ;.      ERROR IN RECORDING HITS IN HIT/MISS
 16798                                  ;ENDIF
 16799                                  ;CLEAR CCR BIT<9>
 16800                                  ;READ FIRST LOCATION
 16801                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
 16802                                  ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
 16803                                  ;ENDIF
 16804                                  ;READ SECOND LOCATION
 16805                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
 16806                                  ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
 16807                                  ;ENDIF
 16808                                  ;ENDTST
 16809                                  ;
 16810                                  ;;**************************************************************
 16811  103442  000004                 TST12:  SCOPE
 16812  103444  005737  003122                 TST     @#TSTLOC                ;ALLOCATE FIRST TEST LOCATION
 16813  103450  005737  003124                 TST     @#TSTLOC+2              ;ALLOCATE SECOND TEST LOCATION
 16814  103454  052737  001000  177746         BIS     #BIT09, CCR             ;SET CCR BIT 9 (CACHE BYPASS)
 16815  103462  005737  003122         1$:     TST     @#TSTLOC                ;READ FIRST TEST LOCATION
 16816  103466  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16817  103474  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
 16818  103502  001001                         BNE     2$                      ;THEN
 16819  103504  104045                         ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<9>
 16820  103506  005037  003124         2$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
 16821  103512  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16822  103520  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
 16823  103526  001001                         BNE     3$                      ;THEN
 16824  103530  104045                         ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCCR<9
 16825  103532  005737  003122         3$:     TST     @#TSTLOC                ;READ FIRST TEST LOCATION
 16826  103536  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16827  103544  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
 16828  103552  001401                         BEQ     4$                      ;THEN
 16829  103554  104025                         ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
 16830  103556  005037  003124         4$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
 16831  103562  013737  177752  110152         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
 16832  103570  032737  000004  110152         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
 16833  103576  001401                         BEQ     5$                      ;THEN
 16834  103600  104025                         ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
 16835  103602  042737  001000  177746 5$:     BIC     #BIT09,CCR              ;RESTORE CCR
 16836
```

TEST - WRITE WRONG DATA PARITY TEST SEQ 0308

```
16837                           .SBTTL  TEST  - WRITE WRONG DATA PARITY TEST
16838                           ;WRITE WRONG DATA PARITY TEST - THIS TEST WILL VERIFY THAT WHEN CCR
16839                           ;BIT<6> = 1 AND CCR BITS<7,0> = 0,1, A READ MISS OCCURS AFTER A
16840                           ;WRITE. THE WRITE WITH CCR BIT<6> = 1 TO A LOCATION WILL CAUSE A
16841                           ;CACHE UPDATE AND WRONG, PARITY TO BE WRITTEN SO WHEN THE LOCATION
16842                           ;IS READ INSTEAD OF A HIT BEING RECORDED THE CACHE PARITY ERROR
16843                           ;WILL CAUSE A CACHE MISS. THIS TEST WILL BE DONE A BYTE AT A TIME.
16844                           ;
16845                           ;BGNTST
16846                           ;SAVE CONTENTS OF VECTOR 114
16847                           ;LET VECTOR 114 POINT TO ABORT ROUTINE
16848                           ;CLEAR MSER
16849                           ;IF MSER NOT CLEAR THEN
16850                           ;.      ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
16851                           ;ENDIF
16852                           ;WRITE TEST LOCATION
16853                           ;SET BITS<6,0> IN CCR
16854                           ;WRITE TEST LOCATION LOW BYTE
16855                           ;CLEAR CCR BIT<6> (WRITE WRONG DATA PARITY)
16856                           ;INITIALIZE ERROR INDICATORS
16857                           ;READ TEST LOCATION LOW BYTE
16858                           ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
16859                           ;.      PARITY ERROR DOESN'T CAUSE MISS
16860                           ;ELSE
16861                           ;.      IF MSER BITS <7:5> ZERO THEN
16862                           ;.      .       PARITY ERROR DOESN'T SET MSER PROPERLY
16863                           ;.      ELSE
16864                           ;.      .       SET ERROR IN LOW BYTE INDICATOR
16865                           ;.      ENDIF
16866                           ;.      ENDIF
16867                           ;ENDIF
16868                           ;CLEAR MSER
16869                           ;IF MSER NOT CLEAR THEN
16870                           ;.      .       ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
16871                           ;ENDIF
16872                           ;SET CCR BIT<6>
16873                           ;WRITE TEST LOCATION HIGH BYTE
16874                           ;CLEAR CCR BIT<6>
16875                           ;READ TEST LOCATI   HIGH BYTE
16876                           ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
16877                           ;.      IF MSER BITS<7:5> NOT SET THEN
16878                           ;.      .       PARITY ERROR DOESN'T CAUSE A MISS
16879                           ;.      ELSE
16880                           ;.      .       SET ERROR IN HIGH BYTE INDICATOR
16881                           ;.      ENDIF
16882                           ;ELSE
16883                           ;.      IF MSER BITS <7:5> ZERO THEN
16884                           ;.      .       PARITY ERROR DOESN'T SET MSER
16885                           ;.      ENDIF
16886                           ;ENDIF
16887                           ;RESTORE CONTENTS OF VECTOR 114
16888                           ;IF ERROR INDICATORS SET THEN
16889                           ;.      IF ERROR IN BOTH BYTES THEN
16890                           ;.      .       PARITY ERROR IGNORED
16891                           ;.      ELSE
16892                           ;.      .       IF ERROR IN LOW BYTE THEN
```

```
16893                                 ;.      .       .       LOW BYTE PARITY ERROR IGNORED
16894                                 ;.      .       ELSE
16895                                 ;.      .       .       HIGH BYTE PARITY ERROR IGNORED
16896                                 ;.      .       ENDIF
16897                                 ;.      ENDIF
16898                                 ;ENDIF
16899                                 ;EXIT   TST
16900                                 ;
16901                                 ;DATA PARITY ABORT ROUTINE:
16902                                 ;                               ILLEGAL PARITY INTERRUPT
16903                                 ;                               RETURN
16904                                 ;
16905                                 ;ENDTST
16906                                 ;;*****************************************************************
16907   103610  000004                TST13:  SCOPE
16908   103612  013737  000114  002762        MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16909   103620  012737  104062  000114        MOV     #DAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
16910   103626  005037  177744              CLR     MSER                    ;CLEAR MSER
16911   103632  005737  177744              TST     MSER                    ;IF MSER NOT CLEAR
16912   103636  001401                      BEQ     1$                      ;THEN
16913   103640  104026                      ERROR   ;26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
16914   103642  005037  003122        1$:   CLR     @#TSTLOC                ;WRITE TEST LOCATION TO ALLOCATE CACHE
16915   103646  012737  000101  177746        MOV     #101,   CCR             ;SET BITS<6,0> IN CCR
16916   103654  112737  000377  003122        MOVB    #377,   TSTLOC          ;WRITE LOW BYTE WITH BAD PARITY
16917   103662  042737  000100  177746        BIC     #BIT06, CCR             ;CLEAR WRITE WRONG DATA PARITY BIT
16918   103670  005002                      CLR     R2                      ;CLEAR ERROR INDICATORS
16919   103672  105737  003122              TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
16920   103676  013703  177752              MOV     HITMIS, R3              ;SAVE HIT/MISS
16921   103702  032703  000004              BIT     #BIT02, R3              ;IF BIT 1 SET IN HIT/MISS REGISTER
16922   103706  001402                      BEQ     2$                      ;THEN
16923   103710  104027                      ERROR   ;27                     ;PARITY ERROR DON'T CAUSE A MISS
16924   103712  000405                      BR      3$                      ;ELSE
16925   103714  032737  000340  177744  2$:   BIT     #340,   MSER            ;IF MSER BIT<7:5> NOT ZERO
16926   103722  001001                      BNE     3$                      ;THEN
16927   103724  104030                      ERROR   ;30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
16928   103726  005037  177744        3$:   CLR     MSER                    ;CLEAR MSER
16929   103732  005737  177744              TST     MSER                    ;IF MSER NOT CLEAR
16930   103736  001401                      BEQ     4$                      ;THEN
16931   103740  104026                      ERROR   ;26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
16932   103742  052737  000100  177746  4$:   BIS     #BIT06, CCR             ;SET CCR BIT 6 (WRITE WRONG PARITY)
16933   103750  112737  000377  003123        MOVB    #377,   @#TSTLOC+1      ;WRITE HIGH BYTE OF TEST LOCATION
16934   103756  042737  000100  177746        BIC     #BIT06, CCR             ;CLEAR WRITE WRONG PARITY BIT
16935   103764  105737  003123              TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
16936   103770  013737  177752  110152        MOV     HITMIS,RECDAT           ;SAVE HIT/MISS
16937   103776  032737  000004  110152        BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
16938   104004  001402                      BEQ     5$                      ;THEN
16939   104006  104027                      ERROR   ;27                     ;PARITY ERROR DON'T CAUSE A MISS
16940   104010  000405                      BR      6$                      ;ELSE
16941   104012  032737  000340  177744  5$:   BIT     #340,   MSER            ;IF BITS <7:5> ZERO
16942   104020  001001                      BNE     6$                      ;THEN
16943   104022  104030                      ERROR   ;30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
16944   104024  005037  177746        6$:   CLR     CCR                     ;CLEAR CCR BEFORE EXIT
16945   104030  013737  002762  000114        MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
16946   104036  032702  000003              BIT     #3,     R2              ;IF ERROR INDICATORS SET
16947   104042  001401                      BEQ     7$                      ;THEN
16948   104044  104031                      ERROR   ;31                     ;PARITY ERROR IGNORED
```

```
COKDAAO KDJ11-B CLUSTER MACY11 30(1046)  23-JAN-84  18:56  PAGE 310
COKDAA.P11    23-JAN-84 18:55           TEST  - WRITE WRONG DATA PARITY TEST                                    SEQ 0310

 16949  104046  005037  177746          7$:     CLR     CCR                             ;CLEAR CCR
 16950  104052  013737  002762  000114          MOV     SLOC00,@#114                    ;RESTORE PARITY TRAP
 16951  104060  000404                          BR      TST14           ;;GO TO NEXT TEST
 16952
 16953
 16954  104062  011637  001122          DAPABO: MOV     (SP),   $BDADR                  ;SAVE ADDRESS THAT CAUSED ABORT
 16955  104066  104007                          ERROR   .7                              ;ILLEGAL PARITY INTERRUPT
 16956  104070  000002                          RTI
 16957
```

```
16958                                   .SBTTL  TEST  - WRITE WRONG TAG PARITY
16959                                   ;WRITE WRONG TAG PARITY - THIS TEST WILL VERIFY THAT A READ MISS
16960                                   ;OCCURS AFTER A WRITE WILL CCR<10> = 1 AND CCR<7,0> = 0,1. THE WRITE
16961                                   ;TO THE LOCATION WILL CAUSE A CACHE UPDATE BUT THE TAG WILL BE
16962                                   ;WRITTEN WITH THE WRONG PARITY. WHEN THE LOCATION IS READ INSTEAD
16963                                   ;OF A CACHE HIT OCCURRING THE PARITY ERROR SHOULD CAUSE A CACHE
16964                                   ;MISS.
16965                                   ;
16966                                   ;BGNTST
16967                                   ;SAVE CONTENTS OF VECTOR 114
16968                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
16969                                   ;CLEAR MSER
16970                                   ;SET WRITE WRONG TAG PARITY BIT<10>
16971                                   ;WRITE TEST LOCATION
16972                                   ;CLEAR CCR BIT<10> AND SET ABORT DISABLE BIT<0>
16973                                   ;READ TEST LOCATION
16974                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
16975                                   ;.      IF MSER BITS <7:5> SET THEN
16976                                   ;.              .       PARITY ERROR DOESN'T CAUSE MISS
16977                                   ;.      ELSE
16978                                   ;.              .       PARITY ERROR DOESN'T SET MSER WITH CCR<7>=0
16979                                   ;.      ENDIF
16980                                   ;ENDIF
16981                                   ;SET WRITE WRONG TAG PARITY BIT<10>
16982                                   ;WRITE TO A BYTE OF A TEST LOCATION
16983                                   ;SAVE HIT/MISS REGISTER
16984                                   ;CLEAR WRITE WRONG TAG PARITY BIT
16985                                   ;IF BIT<1> NOT SET IN HIT/MISS REGISTER THEN
16986                                   ;.      ERROR
16987                                   ;ENDIF
16988                                   ;RESTORE VECTOR 114
16989                                   ;EXIT   TST
16990                                   ;
16991                                   ;TAG PARITY ABORT ROUTINE:
16992                                   ;                               ILLEGAL PARITY INTERRUPT
16993                                   ;                               RETURN
16994                                   ;ENDTST
16995                                   ;;*****************************************************************
16996  104072  000004                   TST14:  SCOPE
16997  104074  013737  000114  002762           MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16998  104102  012737  104174  000114           MOV     #TAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
16999  104110  005037  177744                   CLR     MSER                    ;CLEAR MSER
17000  104114  012737  002000  177746           MOV     #BIT10, CCR             ;SET WRITE WRONG TAG PARITY
17001  104122  005037  003122                   CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD TAG PARITY
17002  104126  012737  000001  177746           MOV     #BIT00, CCR             ;CLEAR BIT 10 AND SET BIT 0
17003  104134  005737  003122                   TST     @#TSTLOC                ;READ TEST LOCATION
17004  104140  013737  177752  110152           MOV     HITMIS,RECDAT           ;SAVE HIT/MISS REGISTER
17005  104146  032737  000004  110152           BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
17006  104154  001401                           BEQ     2$                      ;THEN
17007  104156  104027                           ERROR   ^27                     ;PARITY ERROR DON'T CAUSE A MISS
17008  104160  013737  002762  000114  2$:      MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
17009  104166  005037  177744                   CLR     MSER                    ;CLEAR ERRORS
17010  104172  000404                           BR      TST15           ;;GO TO NEXT TEST
17011
17012
17013  104174  011637  001122           TAPABO: MOV     (SP),@#BDADR            ;SAVE ADDRESS THAT CAUSE ABORT
```

```
 17014  104200  104007                        ERROR    ,7                    ;ILLEGAL PARITY INTERRUPT
 17015  104202  000002                        RTI
 17016
```

```
17017                                   .SBTTL  TEST  - PARITY ABORT TEST
17018                                   ;PARITY ABORT TEST - THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
17019                                   ;1,0, AN ABORT OCCURS ON THE EXECUTION OF AN INSTRUCTION THAT HAS
17020                                   ;BEEN WRITTEN WITH WRONG PARITY.
17021                                   ;
17022                                   ;
17023                                   ;BGNTST
17024                                   ;SAVE VECTOR 114
17025                                   ;SAVE VECTOR 4
17026                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17027                                   ;LET VECTOR 4 POINT TO ERROR ROUTINE
17028                                   ;CLEAR EXPECTING ABORT FLAG
17029                                   ;SET CCR<10> WRITE WRONG TAG PARITY BIT
17030                                   ;WRITE TEST ADDRESS
17031                                   ;CLEAR CCR<10>
17032                                   ;SET CCR TO #200 ENABLE PARITY ABORTS
17033                                   ;SET EXPECTING ABORT FLAG
17034                                   ;READ TEST ADDRESS
17035                                   ;IF ABORT FLAG NE 0 THEN
17036                                   ;.      ERROR IN PARITY ABORT LOGIC
17037                                   ;ENDIF
17038                                   ;RESTORE VECTOR 114
17039                                   ;RESTORE VECTOR 4
17040                                   ;EXIT   TST
17041                                   ;
17042                                   ;ABORT ROUTINE: IF EXPECTING ABORT FLAG NOT SET THEN
17043                                   ;                       .       ERROR NO ABORT SHOULD HAVE OCCURRED
17044                                   ;               ELSE
17045                                   ;                       .       CLEAR (EXPECTING) ABORT FLAG
17046                                   ;               ENDIF
17047                                   ;               IF MSER NOT EQUAL TO 100040 THEN
17048                                   ;                       .       PARITY ABORT LOGIC DOESN'T SET MSER PROPERLY
17049                                   ;               ENDIF
17050                                   ;               IF PC = UPDATED PC THEN
17051                                   ;                       .       ILLEGAL PARITY ABORT
17052                                   ;               ENDIF
17053                                   ;               RETURN
17054                                   ;
17055                                   ;ENDTST
17056                                   ;;*****************************************************************
17057   104204  000004                  TST15:  SCOPE
17058   104206  013737  000114  002762          MOV     @#114,  SLOC00          ;SAVE VECTOR 114
17059   104214  013737  000004  002764          MOV     @#4,    SLOC01          ;SAVE VECTOR 4
17060   104222  012737  104324  000114          MOV     #ABORTR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
17061                                   ;
17062                                   ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
17063                                   ;
17064   104230  012704  104230                  MOV     #.,R4                   ;START WITH CURRENT
17065   104234  005724                  1$:     TST     (R4)+                   ;READ A WORD
17066   104236  022704  104324                  CMP     #ABORTR,R4              ;GOT TO ABORT ROUTINE?
17067   104242  001374                          BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
17068   104244  005000                          CLR     R0                      ;CLEAR EXPECTING ABORT FLAG
17069   104246  012737  002000  177746          MOV     #BIT10, CCR             ;SET WRITE WRONG PARITY BIT
17070   104254  005037  003122                  CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH BAD PARITY
17071   104260  012737  000200  177746          MOV     #BIT07, CCR             ;ENABLE ABORTS, CLEAR WWTP BIT
17072   104266  005100                          COM     R0                      ;SET EXPECTING ABORT FLAG
```

```
17073  104270  005737  003122          ABORTI: TST     @#TSTLOC                ;READ TEST LOCATION (SHOULD CAUSE ABORT)
17074  104274  005700                          TST     R0                      ;IF ABORT FLAG NOT EQUAL ZERO
17075  104276  001401                          BEQ     1$                      ;THEN
17076  104300  104034                          ERROR   .34                     ;PARITY ABORT LOGIC DOESN'T WORK
17077  104302  013737  002762  000114  1$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17078  104310  013737  002764  000004          MOV     SLOC01, @#4             ;RESTORE VECTORE 4
17079  104316  005037  177744                  CLR     MSER
17080  104322  000426                          BR      TST16           ;;GO TO NEXT TEST
17081
17082
17083  104324  013703  177744          ABORTR: MOV     MSER,R3                 ;SAVE MSER
17084  104330  005700                          TST     R0                      ;IF EXPECTING ABORT FLAG NOT SET
17085  104332  001004                          BNE     1$                      ;THEN
17086  104334  011637  001122                  MOV     (SP),   $BDADR          ;SAVE ABORT ADDRESS
17087  104340  104007                          ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17088  104342  000401                          BR      2$                      ;ELSE
17089  104344  005000                  1$:     CLR     R0                      ;CLEAR (EXPECTING) ABORT FLAG
17090  104346  022737  100040  177744  2$:     CMP     #100040,MSER            ;IF MSER NOT EQUAL TO 100040
17091  104354  001404                          BEQ     3$                      ;THEN
17092  104356  012737  100040  001124          MOV     #100040,$GDDAT          ;SAVE PROPER MSER SETTING
17093  104364  104035                          ERROR   .35                     ;PARITY ABORT DON'T SET MSER PROPERLY
17094  104366  021627  104274          3$:     CMP     (SP),   #ABORTI+4       ;IF PC EQUAL TO UPDATE PC
17095  104372  001401                          BEQ     4$                      ;THEN
17096  104374  104007                          ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17097  104376  000002                  4$:     RTI                             ;RETURN
17098
```

```
                                .SBTTL  TEST  - PARITY INTERRUPT TEST
                                ;PARITY INTERRUPT TEST - THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
                                ;0,0, A PARITY INTERRRUPT OCCURS AFTER EXECUTION OF AN INSTRUCTION
                                ;THAT HAS BEEN WRITTEN WITH WRONG PARITY.
                                ;
                                ;BGNTST
                                ;SAVE CONTENTS OF 114
                                ;SETUP VECTOR 114 TO POINT TO INTERRUPT ROUTINE
                                ;CLEAR EXPECTING INTERRUPT FLAG
                                ;WRITE TEST ADDRESS WITH BAD PARITY
                                ;SET EXPECTING INTERRUPT FLAG
                                ;READ TEST ADDRESS
                                ;IF INTERRUPT FLAG NE 0 THEN
                                ;.      PARITY INTERRUPT LOGIC DOESN'T WORK
                                ;ENDIF
                                ;RESTORE CONTENTS OF VECTOR 114
                                ;EXIT   TST
                                ;
                                ;INTERRUPT ROUTINE:     IF EXPECTING INTERRUPT FLAG NE 1 THEN
                                ;                       .       ERROR NO INTERRUPT SHOULD HAVE OCCURRED
                                ;                       ELSE
                                ;                       .       CLEAR (EXPECTING) INTERRUPT FLAG
                                ;                       ENDIF
                                ;                       IF SAVED PC NE TO UPDATED PC THEN
                                ;                       .       IF PC = TEST INSTRUCTION PC THEN
                                ;                       .       .       ERROR INSTRUCTION ABORTED
                                ;                       .       ELSE
                                ;                       .       .       ILLEGAL PARITY ABORT
                                ;                       .       ENDIF
                                ;                       ENDIF
                                ;                       IF MSER NE #340 THEN
                                ;                       .       PARITY INTERRUPT DOESN'T SET MSER PROPERLY
                                ;                       ENDIF
                                ;                       RETURN
                                ;
                                ;ENDTST
                                ;;*****************************************************************
104400  000004                  TST16:  SCOPE
104402  013737  000114  002762          MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
104410  012737  104476  000114          MOV     #INTERR,@#114           ;LET VECTOR POINT TO INTERRUPT ROUTINE
                                ;
                                ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
                                ;
104416  012704  104416          1$:     MOV     #.,R4                   ;START WITH CURRENT
104422  005724                  1$:     TST     (R4)+                   ;READ A WORD
104424  022704  104476                  CMP     #INTERR,R4              ;GOT TO INTERRUPT ROUTINE?
104430  001374                          BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
104432  005001                          CLR     R1                      ;CLEAR EXPECTING INTERRUPT FLAG
104434  052737  000100  177746          BIS     #BIT06, CCR             ;SET WRITE WRONG DATA PARITY
104442  005037  003122                  CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD DATA PARITY
104446  005037  177746                  CLR     CCR                     ;CLEAR WRITE WRONG DATA PARITY
104452  005101                          COM     R1                      ;SET EXPECTING INTERRUPT FLAG
104454  005737  003122          INTRPC: TST     @#TSTLOC                ;READ TEST LOCATION
104460  005701                          TST     R1                      ;IF INTERRUPT FLAG NOT EQUAL ZERO
104462  001401                          BEQ     1$                      ;THEN
104464  104036                          ERROR   .36                     ;PARITY INTERRUPT LOGIC DOESN'T WORK
```

```
17155  104466  013737  002762  000114  1$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17156  104474  000424                          BR      TST17           ;;GO TO NEXT TEST
17157
17158
17159  104476  005701                  INTERR: TST     R1                      ;IF EXPECTING INTERRUPT FLAG NOT SET
17160  104500  001004                          BNE     1$                      ;THEN
17161  104502  011637  001122                  MOV     (SP),   $BDADR          ;SAVE INTERRUPT ADDRESS
17162  104506  104007                          ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17163  104510  000401                          BR      2$                      ;ELSE
17164  104512  005001                  1$:     CLR     R1                      ;CLEAR (EXPECTING) INTERRUPT FLAG
17165  104514  021627  104460          2$:     CMP     (SP),   #INTRPC+4       ;IF SAVED PC NOT EQUAL TO UPDATED PC
17166  104520  001401                          BEQ     4$                      ;THEN
17167  104522  104007                          ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17168  104524  022737  000340  177744  4$:     CMP     #340,   MSER            ;IF MSER NOT EQUAL TO EXPECTED VALUE
17169  104532  001404                          BEQ     5$                      ;THEN
17170  104534  012737  000340  001124          MOV     #340,   $GDDAT          ;SAVE PROPER MSER
17171  104542  104035                          ERROR   .35                     ;PARITY INTERRUPT DON'T SET MSER PROPERLY
17172  104544  000002                  5$:     RTI                             ;RETURN
17173
```

```
17174                                   .SBTTL  TEST    - MISCELLANEOUS PARITY TEST
17175                                   ;MISCELLANEOUS PARITY TEST - THIS TEST CHECKS THAT BYPASS CYCLES WITH
17176                                   ;PARITY ERRORS CAUSE CACHE HIT RESPONSE AND THAT FORCE MISS CYCLES
17177                                   ;IGNORE PARITY ERRORS.
17178                                   ;
17179                                   ;BGNTST
17180                                   ;WRITE A LOCATION WITH BAD PARITY
17181                                   ;SET BYPASS IN CCR
17182                                   ;READ THE LOCATION BACK
17183                                   ;IF NO HIT OR MSER NOT SET THEN
17184                                   ;.      ERROR
17185                                   ;ENDIF
17186                                   ;WRITE A LOCATION WITH BAD PARITY AND SET BYPASS
17187                                   ;IF MSER SET WHILE READING IT BACK
17188                                   ;.      ERROR
17189                                   ;ENDIF
17190                                   ;ENDTST
17191                                   ;;*****************************************************************
17192  104546  000004                  TST17:  SCOPE
17193  104550  012703  104550                  MOV     #.,R3                   ;START WITH CURRENT INSTRUCTION
17194  104554  005723                  10$:    TST     (R3)+                   ;READ A WORD
17195  104556  022703  104720                  CMP     #4$,R3                  ;LAST WORD?
17196  104562  001374                          BNE     10$                     ;IF NOT, CONTINUE
17197                                   ;
17198                                   ; CHECK BYPASS AND BAD PARITY
17199                                   ;
17200  104564  013737  000114  002762          MOV     @#114,  SLOC00          ;SAVE PARITY VECTOR
17201  104572  012737  104626  000114          MOV     #1$,@#114               ;POINT NEW VECTOR
17202  104600  052737  002101  177746          BIS     #BIT10!BIT06!BIT00,CCR  ;DATA AND TAG PAR., NO INT.
17203  104606  005037  003122                  CLR     @#TSTLOC                ;WRITE CYCLE
17204  104612  012737  001000  177746          MOV     #BIT09,CCR              ;SET BYPASS
17205  104620  005737  003122                  TST     @#TSTLOC                ;BYPASS WITH WRONG PARITY
17206  104624  000410                          BR      2$                      ;
17207  104626  062706  000004          1$:     ADD     #4,SP                   ;ADJUST STACK
17208  104632  104045                          ERROR   .45                     ;ERROR
17209  104634  032737  000340  177744          BIT     #340,MSER               ;MSER OK?
17210  104642  001001                          BNE     2$                      ;IF YES, BRANCH
17211  104644  104042                          ERROR   .42                     ;BYPASS WRONG
17212                                   ;
17213                                   ; CHECK FORCE MISS AND BAD PARITY
17214                                   ;
17215  104646  005037  177744          2$:     CLR     MSER                    ;CLEAR MSER
17216  104652  005037  177746                  CLR     CCR                     ;CLEAR CCR
17217  104656  012737  104712  000114          MOV     #3$,@#114               ;POINT NEW VECTOR
17218  104664  052737  002101  177746          BIS     #BIT10!BIT06!BIT00,CCR  ;DATA AND TAG PAR., NO INTER
17219  104672  005037  003122                  CLR     @#TSTLOC                ;FORCE MISS WITH PARITY
17220  104676  012737  000014  177746          MOV     #14,CCR                 ;FORCE MISS
17221  104704  005737  003122                  TST     @#TSTLOC                ;ALLOCATE CACHE
17222  104710  000403                          BR      4$
17223  104712  104042                  3$:     ERROR   .42
17224  104714  062706  000004                  ADD     #4,SP                   ;ADJUST STACK
17225  104720  005037  177746          4$:     CLR     CCR                     ;CLEAR CCR
17226  104724  013737  002762  000114          MOV     SLOC00,@#114            ;RESTORE PARITY VECTOR
17227  104732  012737  000400  177746          MOV     #BIT08,CCR              ;FLUSH CACHE
17228
```

```
17229                                   .SBTTL  TEST  - MEMORY SYSTEM ERROR REGISTER TEST
17230                                   ;MEMORY SYSTEM ERROR REGISTER TEST - THIS TEST WILL VERIFY THE
17231                                   ;FUNCTIONALITY OF BITS <15> AND <7:5> OF THE MEMORY SYSTEM ERROR
17232                                   ;REGISTER. THIS TEST WILL USE THE WRITE WRONG PARITY BITS (BITS<10>
17233                                   ;AND <6> OF THE CCR) TO WRITE BAD PARITY INTO A LOCATION. THE
17234                                   ;LOCATION WILL THEN BE READ WITH CACHE TRAPS ENABLED AND THE MSER
17235                                   ;WILL BE CHECKED AFTER THE ABORT FOR THE CORRECT BIT(S) BEING SET.
17236                                   ;THIS WILL BE DONE FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR WORD
17237                                   ;ACCESSES THEN REPEATED FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR
17238                                   ;BYTE ACCESSES. THE TEST WILL THEN BE REPEATED A THIRD TIME FOR BYTE
17239                                   ;ACCESSES AND ABORTS DISABLED. THE MSER SHOULD CONTAIN BITS <7:5>
17240                                   ;SET TO 1'S AND BIT <15> A ZERO FOR ALL COMBINATIONS.
17241                                   ;
17242                                   ;BGNTST
17243                                   ;SAVE CONTENTS OF VECTOR 114
17244                                   ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17245                                   ;INITIALIZE LOOP COUNTER
17246                                   ;DO UNTIL ALL WORD COMBINATIONS CHECKED
17247                                   ;.      INITIALIZE MSER
17248                                   ;.      SETUP CCR FROM CCR TABLE
17249                                   ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17250                                   ;.      CLEAR <10,6> FROM CCR
17251                                   ;.      SETUP CCR FOR ABORT
17252                                   ;.      READ TEST LOCATION                      ;THIS COULD CAUSE TRAP
17253                                   ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
17254                                   ;.      .       ERROR IN SETTING MSER
17255                                   ;.      ENDIF
17256                                   ;.      UPDATE LOOP COUNTER
17257                                   ;ENDDO
17258                                   ;INITIALIZE LOOP COUNTER
17259                                   ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17260                                   ;.      INITIALIZE MSER
17261                                   ;.      SETUP CCR FROM CCR TABLE
17262                                   ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17263                                   ;.      CLEAR <10,6> FROM CCR
17264                                   ;.      SETUP CCR FOR ABORT
17265                                   ;.      READ LOW BYTE TEST LOCATION
17266                                   ;.      GET EXPECTED BYTE DATA FROM TABLE
17267                                   ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
17268                                   ;.      .       ERROR IN SETTING MSER
17269                                   ;.      ENDIF
17270                                   ;.      INITIALIZE MSER
17271                                   ;.      READ HIGH BYTE TEST LOCATION
17272                                   ;.      GET EXPECTED BYTE DATA FROM TABLE
17273                                   ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
17274                                   ;.      .       ERROR IN SETTING MSER
17275                                   ;.      ENDIF
17276                                   ;.      INCREMENT LOOP COUNTER
17277                                   ;ENDDO
17278                                   ;INITIALIZE LOOP COUNTER
17279                                   ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17280                                   ;.      INITIALIZE MSER
17281                                   ;.      SETUP CCR FROM CCR TABLE
17282                                   ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17283                                   ;.      CLEAR <10,6> FROM CCR
17284                                   ;.      SETUP CCR FOR NO ABORTS
```

```
17285                                   ;.      READ LOW BYTE TEST LOCATION
17286                                   ;.      IF RECEIVED DATA NE TO #340 THEN
17287                                   ;.      .       ERROR IN SETTING MSER
17288                                   ;.      ENDIF
17289                                   ;.      INITIALIZE MSER
17290                                   ;.      READ HIGH BYTE TEST LOCATION
17291                                   ;.      IF RECEIVED DATA NE #340 THEN
17292                                   ;.      .       ERROR IN SETTING MSER
17293                                   ;.      ENDIF
17294                                   ;.      INCREMENT LOOP COUNTER
17295                                   ;ENDDO
17296                                   ;EXIT   TST
17297                                   ;
17298                                   ;CCR TABLE:     0
17299                                   ;               100
17300                                   ;               2000
17301                                   ;               2100
17302                                   ;EXPECTED WORD DATA:    0
17303                                   ;                       100300
17304                                   ;                       100040
17305                                   ;                       100340
17306                                   ;EXPECTED BYTE DATA:    0
17307                                   ;                       0
17308                                   ;                       100100
17309                                   ;                       100200
17310                                   ;                       100040
17311                                   ;                       100040
17312                                   ;                       100140
17313                                   ;                       100240
17314                                   ;ABORT ROUTINE: RTI
17315                                   ;
17316                                   ;ENDTST
17317                                   ;;*****************************************************************
17318  104740  000004                   TST20:  SCOPE
17319  104742  013737  000114  002762           MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17320  104750  012737  105420  000114           MOV     #ABROUT,@#114           ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17321  104756  012704  000004                   MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17322  104762  012700  105360                   MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17323  104766  012701  105370                   MOV     #EXPWDT,R1              ;GET ADDRESS OF EXPECTED DATA TABLE
17324  104772  005037  177744           1$:     CLR     MSER                    ;INITIALIZE MSER DATA
17325  104776  012037  177746                   MOV     (R0)+,  CCR             ;SETUP CCR FROM CCR TABLE
17326  105002  005037  003122                   CLR     @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17327  105006  012737  000200  177746           MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17328  105014  005737  003122                   TST     @#TSTLOC                ;READ TEST LOCATION
17329  105020  021137  177744                   CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17330  105024  001403                           BEQ     2$                      ;DATA THEN
17331  105026  011137  001124                   MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17332  105032  104035                           ERROR   +35                     ;MSER NOT SET PROPERLY
17333  105034  005721                   2$:     TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17334  105036  005737  023122                   TST     @#TSTLOC+8192.          ;TO INSURE MISS ON THE NEXT LOOP
17335  105042  077425                           SOB     R4,     1$              ;LOOP UNTIL ALL COMBINATIONS CHECKED
17336                                   ;CHECK BYTE OPERATIONS NOW
17337  105044  012704  000004                   MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17338  105050  012700  105360                   MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17339  105054  012701  105400                   MOV     #EXPBDT,R1              ;GET ADDRESS OF EXPECTED BYTE DATA TABLE
17340  105060  005037  177744           3$:     CLR     MSER                    ;INITIALIZE MSER
```

```
17341  105064  005737  003122                TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17342  105070  011037  177746                MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17343  105074  105037  003122                CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17344  105100  012737  000200  177746        MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17345  105106  105737  003122                TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17346  105112  021137  177744                CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17347  105116  001403                        BEQ     4$                      ;THEN
17348  105120  011137  001124                MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17349  105124  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17350  105126  005721                4$:     TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17351  105130  005037  177744                CLR     MSER                    ;INITIALIZE MSER
17352  105134  005737  003122                TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17353  105140  012037  177746                MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17354  105144  105037  003123                CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17355  105150  012737  000200  177746        MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17356  105156  105737  003123                TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17357  105162  021137  177744                CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17358  105166  001403                        BEQ     5$                      ;THEN
17359  105170  011137  001124                MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17360  105174  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17361  105176  005721                5$:     TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17362  105200  077451                        SOB     R4,     3$              ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17363                                ;REPEAT WITHOUT ABORT
17364  105202  012704  000003                MOV     #3,     R4              ;INITIALIZE LOOP COUNTER
17365  105206  012700  105362                MOV     #CCRTBL+2,R0            ;GET ADDRESS OF CCR TABLE
17366  105212  005037  177744        6$:     CLR     MSER                    ;INITIALIZE MSER
17367  105216  005737  003122                TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17368  105222  011037  177746                MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17369  105226  105037  003122                CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17370  105232  012737  000001  177746        MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17371  105240  105737  003122                TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17372  105244  022737  000340  177744        CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17373  105252  001404                        BEQ     7$                      ;THEN
17374  105254  012737  000340  001124        MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17375  105262  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17376  105264  005037  177744        7$:     CLR     MSER                    ;INITIALIZE MSER
17377  105270  005737  003122                TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17378  105274  012037  177746                MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17379  105300  105037  003123                CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17380  105304  012737  000001  177746        MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17381  105312  105737  003123                TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17382  105316  022737  000340  177744        CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17383  105324  001404                        BEQ     8$                      ;THEN
17384  105326  012737  000340  001124        MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17385  105334  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17386  105336  077453                8$:     SOB     R4,     6$              ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17387  105340  005037  177744                CLR     MSER                    ;CLEAR ERROR REGISTER
17388  105344  013737  002762  000114        MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17389  105352  005037  177746                CLR     CCR                     ;CLEAR ALL BIT IN CCR
17390  105356  000421                        BR      TST21           ;;GO TO NEXT TEST
17391
17392
17393  105360  000000        CCRTBL: .WORD   0
17394  105362  000100                .WORD   100
17395  105364  002000                .WORD   2000
17396  105366  002100                .WORD   2100
```

```
 17397
 17398  105370  000000                  EXPWDT: .WORD   0
 17399  105372  100300                          .WORD   100300
 17400  105374  100040                          .WORD   100040
 17401  105376  100340                          .WORD   100340
 17402
 17403  105400  000000                  EXPBDT: .WORD   0
 17404  105402  000000                          .WORD   0
 17405  105404  100100                          .WORD   100100
 17406  105406  100200                          .WORD   100200
 17407  105410  100040                          .WORD   100040
 17408  105412  100040                          .WORD   100040
 17409  105414  100140                          .WORD   100140
 17410  105416  100240                          .WORD   100240
 17411
 17412  105420  000002                  ABROUT: RTI
 17413
 17414
```

```
                                .SBTTL  TEST  - CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT
                                ;CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT - THIS TEST WILL
                                ;VERIFY THAT IF A PARITY ERROR OCCURS ON THE SAME ADDRESS REFERENCE AS A
                                ;NON-EXISTENT MEMORY ERROR THAT THE CACHE DATA PATH GATE ARRAY BLOCKS THE
                                ;PARITY ERROR TO THE J-11 CHIP SET. THIS WILL BE DONE BY USING THE DIAGNOSTIC
                                ;BIT TO CAUSE A PARITY ERROR IN A CACHE REFERENCE THAT DOES NOT HAVE A
                                ;CORRESPONDING ADDRESS IN MAIN MEMORY. THE ADDRESS WILL THEN BE READ CAUSING
                                ;BOTH A CACHE PARITY ERROR AND A NON-EXISTENT MEMORY ERROR. TO AVOID HAVING
                                ;TO SIZE THE ENTIRE MEMORY TO FIND A NON-EXISTENT MEMORY ADDRESS THIS TEST
                                ;WILL USE THE LARGEST NON-I/O ADDRESS (17757776). THE TEST WILL FIRST READ
                                ;THIS ADDRESS AND IF A NXM TRAP OCCURS THE TEST WILL BE DONE. IF THE ACCESS
                                ;TO THIS ADDRESS DOES NOT TRAP THEN THE TEST WILL BE SKIPPED.
                                ;
                                ;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 ABORT LOGIC
                                ;ENDIF
                                ;B:
                                ;CLEAR CCR
                                ;RESTORE CONTENTS OF VECTOR 114
                                ;RESTORE CONTENTS OF VECTOR 4
                                ;EXIT   TST
                                ;
                                ;
                                ;NXM-PARITY ERROR ROUTINE:      RESET STACK AFTER TRAP
                                ;                                       PARITY ABORT NOT BLOCKED BY NXM
                                ;                               GOTO B:
                                ;ENDTST
                                ;;*****************************************************************
 105422  000004                 TST21:  SCOPE
 105424  013737  000004  002762         MOV     @#4,    SLOC00          ;SAVE CONTENTS OF VECTOR 4
 105432  013737  000114  002764         MOV     @#114,  SLOC01          ;SAVE CONTENTS OF VECTOR 114
 105440  012737  105466  000004         MOV     #1$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
 105446  012737  177400  172354         MOV     #177400,KIPAR6          ;LET PAR6 = OFFSET TO HIGHEST MEMORY
 105454  005237  177572                 INC     SR0                     ;TURN ON MMU
 105460  005737  157776                 TST     @#157776                ;ACCESS ADDRESS 17757776
 105464  000426                         BR      ABOEXT                  ;IF NO TRAP SKIP TEST
 105466  062706  000004         1$:     ADD     #4,     SP              ;RESET STACK AFTER TRAP
 105472  012737  000102  177746         MOV     #102,   CCR             ;SET DIAG. AND WRITE WRONG PARITY BITS
 105500  005037  157776                 CLR     @#157776                ;WRITE TO ADDRESS 17757776
```

M9


```
 17471  105504  012737  000200  177746          MOV     #200,   CCR             ;CLEAR CCR AND SET PARITY ABORT BIT
 17472  105512  012737  105536  000004          MOV     #2$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
 17473  105520  012737  105570  000114          MOV     #NXMPAR,@#114           ;LET VECTOR 114 POINT TO ERROR ROUTINE
 17474  105526  005737  157776                  TST     @#157776                ;READ ADDRESS 17757776 (SHOULD TRAP)
 17475  105532  104037                          ERROR   .37                     ;NXM AND PARITY ABORT DIN'T HAPPEN
 17476  105534  000402                          BR      ABOEXT                  ;GO EXIT TEST
 17477  105536  062706  000004          2$:     ADD     #4,     SP              ;RESET STACK AFTER TRAP
 17478  105542  005037  177746          ABOEXT: CLR     CCR                     ;CLEAR CCR FOR EXIT
 17479  105546  005037  177572                  CLR     SR0                     ;DISABLE MMU
 17480  105552  013737  002762  000004          MOV     SLOC00, @#4             ;RESTORE VECTOR 4
 17481  105560  013737  002764  000114          MOV     SLOC01, @#114           ;RESTORE VECTOR 114
 17482  105566  000404                          BR      TST22           ;;GO TO NEXT TEST
 17483
 17484
 17485
 17486  105570  062706  000004          NXMPAR: ADD     #4,     SP              ;RESET STACK AFTER TRAP
 17487  105574  104040                          ERROR   .40                     ;PARITY ABORT NOT BLOCKED BY NXM TRAP
 17488  105576  000002                          RTI
 17489
```

```
                                        .SBTTL  TEST  - MULTIPROCCESSING INSTRUCTION TESTS
                                        ;MULTIPROCCESSING INSTRUCTION TESTS - THIS TEST WILL VERIFY THAT THE MULTI-
                                        ;PROCCESSING INSTRUCTIONS DO A BYPASS OF THE CACHE. THIS TEST WILL NOT VERIFY
                                        ;THE REST OF THE FUNCTIONALITY OF THESE INSTRUCTIONS BECAUSE THAT WILL ALREADY
                                        ;HAVE BEEN CHECKED IN THE BASE INSTRUCTION TESTS. THE TEST WILL FIRST ALLOCATE
                                        ;AN ADDRESS IN CACHE THEN A TSTSET INSTRUCTION WILL BE DONE AT THAT ADDRESS.
                                        ;A HIT SHOULD BE RECORDED ON THE ACCESS. NEXT, THE ADDRESS WILL BE READ AND
                                        ;A MISS SHOULD BE RECORDED BECAUSE THE FORCED BYPASS ON THE TSTSET INSTRUCTION
                                        ;SHOULD HAVE INVALIDATED THE CACHE ENTRY. THE SAME SEQUENCE WILL THEN BE
                                        ;REPEATED FOR THE WRTLCK INSTRUCTION.
                                        ;
                                        ;BGNTST
                                        ;READ TEST LOCATION TO ALLOCATE CACHE
                                        ;DO TSTSET INSTRUCTION
                                        ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
                                        ;.      ERROR IN MULTI-PROCESSOR HOOKS
                                        ;ENDIF
                                        ;READ TEST LOCATION (ALSO ALLOCATES CACHE FOR WRTLCK)
                                        ;DO WRTLCK INSTRUCTION
                                        ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
                                        ;.      ERROR IN MULTI-PROCESSOR HOOKS
                                        ;ENDIF
                                        ;READ TEST LOCATION
                                        ;DO ASRB INSTRUCTION
                                        ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
                                        ;.      ERROR IN MULTI-PROCESSOR HOOKS
                                        ;ENDIF
                                        ;ENDTST
                                        ;NOTE: THE CODE IS POSITION DEPENDENT

                                        ;;**********************************************************************
105600  000004                  TST22:  SCOPE
105602  005737  003122                  TST     TSTLOC                  ;READ TEST LOCATION TO ALLOCATE CACHE
                                        ;       TSTSET  TSTLOC                  ;DO TSTSET INSTRUCTION
105606  007237                  7$:     .WORD   7237                    ;THESE NEXT TWO LOCATIONS ARE THE TSTSET
105610  003122                          .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
105612  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
105620  032737  000010  110152          BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
105626  001001                          BNE     1$                      ;THEN
105630  104045                          ERROR   ^45                     ;ERROR IN RECORDING HITS IN HIT/MISS
105632  032737  000004  110152  1$:     BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
105640  001404                          BEQ     2$                      ;THEN
105642  013737  105606  001126          MOV     @#7$,   $BDDAT          ;SAVE OPCODE
105650  104041                          ERROR   ^41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE
                                        ;2$:    WRTLCK  TSTLOC                  ;DO WRTLCK INSTRUCTION
105652  007337                  2$:     .WORD   7337                    ;THESE NEXT TWO WORDS ARE THE WRTLCK
105654  003122                          .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
105656  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
105664  032737  000010  110152          BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
105672  001001                          BNE     3$                      ;THEN
105674  104045                          ERROR   ^45                     ;ERROR IN RECORDING HITS IN HIT/MISS
105676  032737  000004  110152  3$:     BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
105704  001404                          BEQ     4$                      ;THEN
105706  013737  105652  001126          MOV     @#2$,   $BDDAT          ;SAVE OPCODE
105714  104041                          ERROR   ^41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE
105716  106237  003122          4$:     ASRB    TSTLOC                  ;THE THIRD BYPASS INSTRUCTION
```

```
 17546  105722  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
 17547  105730  032737  000020  110152          BIT     #BIT4,RECDAT            ;IF HIT/MISS REGISTER BIT 4 NOT SET
 17548  105736  001001                          BNE     5$                      ;THEN
 17549  105740  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 17550  105742  032737  000010  110152  5$:     BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 SET
 17551  105750  001404                          BEQ     TST23           ;;EXIT TEST
 17552  105752  013737  105716  001126          MOV     @#4$,   $BDDAT          ;SAVE OPCODE
 17553  105760  104041                          ERROR   .41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE
 17554
 17555
 17556
```

```
                                        .SBTTL  TEST  - DATA STORE RAM TESTS
                                        ;DATA STORE RAM TESTS   THERE ARE TWO TESTS FOR THE DATA STORE RAM.
                                        ;THE FIRST TEST WILL BE A NO DUAL ADDRESSING TEST AND THE SECOND
                                        ;TEST WILL BE A DATA RELIABILITY TEST. THE NO DUAL ADDRESSING TEST
                                        ;WILL FIRST WRITE EACH WORD OF THE CACHE RAM WITH ITS WORD ADDRESS
                                        ;AND VERIFY THE CONTENTS WITH A READ OF EACH ADDRESS. THEN EACH
                                        ;BYTE WILL BE WRITTEN WITH ITS ADDRESS AND CHECKED. THE DATA
                                        ;RELIABILITY TEST THAT WILL BE USED IS A TEST CALLED MOVING INVERSIONS.
                                        ;
                                        ;BGNTST 1
                                        ;SETUP MMU REGISTERS TO HAVE BYPASS ON KERNAL SPACE AND NO
                                        ;       BYPASS ON USER SPACE
                                        ;LET PS EQUAL KERNAL FOR CURRENT MODE AND USER FOR PREVIOUS
                                        ;       MODE
                                        ;ENABLE MMU
                                        ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
                                        ;CLEAR DATA TO BE WRITTEN
                                        ;SET DIAGNOSTIC BIT (BIT<1>) IN CCR
                                        ;DO UNTIL DATA TO BE WRITTEN EQUALS 20000(8)
                                        ;.      WRITE DATA TO ADDRESS
                                        ;.      ADD 2 TO ADDRESS
                                        ;.      ADD 2 TO DATA TO BE WRITTEN
                                        ;ENDDO
                                        ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
                                        ;CLEAR EXPECTED DATA
                                        ;DO UNTIL EXPECTED DATA EQUALS 20000(8)
                                        ;.      READ ADDRESS
                                        ;.      IF RECEIVED DATA NE EXPECTED DATA THEN
                                        ;.              GOTO DATA STORE PARITY ERROR ROUTINE
                                        ;.      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
                                        ;.              GOTO DATA STORE PARITY ERROR ROUTINE
                                        ;.      ENDIF
                                        ;.      ADD 1 TO EXPECTED DATA
                                        ;.      ADD 1 TO ADDRESS
                                        ;ENDDO
                                        ;ENDTST
                                        ;
                                        ;;*****************************************************************
17610   105762  000004                  TST23:  SCOPE
17611   105764  004737  131362                  JSR     PC,     INITMM          ;SETUP MMU REGISTERS
17612   105770  012737  002000  172354          MOV     #2000,  KIPAR6          ;LET PAR6 MAP TO ADDRESS 200000
```

```
 17613  105776  005237  177572                      INC     SR0             ;TURN ON MEMORY MANAGEMENT
 17614  106002  012702  140000                      MOV     #140000,R2      ;GET FIRST ADDRESS OF 4K WORD BUFFER
 17615  106006  005001                              CLR     R1              ;INIT DATA TO BE WRITTEN
 17616  106010  052737  000002  177746              BIS     #BIT01, CCR     ;SET DIAG BIT IN CASE NO MEMORY PRESENT
 17617  106016  020127  020000          1$:         CMP     R1,     #20000  ;DO UNTIL ALL ADDRESSES WRITTEN
 17618  106022  001404                              BEQ     2$              ;IF DONE GO CHECK DATA
 17619  106024  010122                              MOV     R1,     (R2)+   ;WRITE DATA TO ADDRESS
 17620  106026  062701  000002                      ADD     #2,     R1      ;UPDATE DATA BY 2 (WORD BOUNDARY)
 17621  106032  000771                              BR      1$              ;ENDDO
 17622  106034  042737  000002  177746  2$:         BIC     #BIT01, CCR     ;CLEAR DIAGNOSTIC BIT
 17623  106042  012702  140000                      MOV     #140000,R2      ;GET FIRST ADDRESS OF 4K WORD BUFFER
 17624  106046  005001                              CLR     R1              ;INIT DATA TO BE CHECKED
 17625  106050  020127  020000          3$:         CMP     R1,     #20000  ;DO UNTIL ALL ADDRESSES CHECKED
 17626  106054  001426                              BEQ     5$              ;IF DONE GO DO BYTES
 17627  106056  010137  001122                      MOV     R1,     $BDADR  ;STORE FOR ERRORS
 17628  106062  052737  100000  001122              BIS     #BIT15, $BDADR  ;
 17629  106070  012237  110152                      MOV     (R2)+,  RECDAT  ;READ TEST LOCATION
 17630  106074  013737  177752  001160              MOV     HITMIS, $TMP0   ;STORE REGISTER
 17631  106102  032737  000020  001160              BIT     #BIT04,$TMP0    ;HIT?
 17632  106110  001001                              BNE     100$            ;IF YES, BRANCH
 17633  106112  104045                              ERROR   ^45             ;DATA RAM ERROR
 17634  106114  020137  110152          100$:       CMP     R1,     RECDAT  ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
 17635  106120  001401                              BEQ     4$              ;THEN
 17636  106122  104043                              ERROR   ^43             ;DATA RAM ERROR
 17637  106124  062701  000002          4$:         ADD     #2,     R1      ;UPDATE EXPECTED DATA BY 2
 17638  106130  000747                              BR      3$              ;ENDDO
 17639  106132  012702  140000          5$:         MOV     #140000,R2      ;GET FIRST ADDRESS OF 4K WORD BUFFER
 17640  106136  005001                              CLR     R1              ;INIT DATA TO BE WRITTEN
 17641  106140  052737  000002  177746              BIS     #BIT01, CCR     ;SET DIAG BIT IN CASE NO MEMORY PRESENT
 17642  106146  020127  020000          6$:         CMP     R1,     #20000  ;DO UNTIL ALL ADDRESSES WRITTEN
 17643  106152  001404                              BEQ     7$              ;IF DONE GO CHECK DATA
 17644  106154  110122                              MOVB    R1,     (R2)+   ;WRITE DATA TO ADDRESS
 17645  106156  062701  000001                      ADD     #1,     R1      ;UPDATE DATA BY 1 (BYTE BOUNDARY)
 17646  106162  000771                              BR      6$              ;ENDDO
 17647  106164  042737  000002  177746  7$:         BIC     #BIT01, CCR     ;CLEAR DIAGNOSTIC BIT
 17648  106172  012702  140000                      MOV     #140000,R2      ;GET FIRST ADDRESS OF 4K WORD BUFFER
 17649  106176  005001                              CLR     R1              ;INIT DATA TO BE CHECKED
 17650  106200  020127  020000          8$:         CMP     R1,     #20000  ;DO UNTIL ALL ADDRESSES CHECKED
 17651  106204  001426                              BEQ     10$             ;IF DONE EXIT TEST
 17652  106206  010137  001122                      MOV     R1,     $BDADR  ;STORE FOR ERRORS
 17653  106212  052737  100000  001122              BIS     #BIT15, $BDADR  ;
 17654  106220  112237  110152                      MOVB    (R2)+,  RECDAT  ;READ TEST LOCATION
 17655  106224  013737  177752  001160              MOV     HITMIS, $TMP0   ;STORE REGISTER
 17656  106232  032737  000020  001160              BIT     #BIT04,$TMP0    ;HIT?
 17657  106240  001001                              BNE     101$            ;IF YES, BRANCH
 17658  106242  104045                              ERROR   ^45             ;DATA RAM ERROR
 17659  106244  120137  110152          101$:       CMPB    R1,     RECDAT  ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
 17660  106250  001401                              BEQ     9$              ;THEN
 17661  106252  104043                              ERROR   ^43             ;DATA RAM ERROR
 17662  106254  062701  000001          9$:         ADD     #1,     R1      ;UPDATE EXPECTED DATA BY 1
 17663  106260  000747                              BR      8$              ;ENDDO
 17664  106262  005037  177572          10$:        CLR     SR0             ;TURN OFF MMU
 17665
```

```
17666                                   .SBTTL  TEST  - TAG STORE RAM TESTS
17667                                   ;TAG STORE RAM TESTS - THERE ARE TWO TESTS FOR THE TAG STORE RAMS. THE FIRST
17668                                   ;TEST IS AN ADDRESSING TEST TO ENSURE NO DUAL ADDRESSING OF THE TAG STORE.
17669                                   ;THE SECOND TEST IS A "MOVING INVERSIONS" TEST THAT CHECKS THE DATA RELIABILITY
17670                                   ;OF THE RAM STORE.
17671                                   ;
17672                                   ;DUAL ADDRESSING TEST - THIS WILL BE DONE BY FIRST FLUSHING THE CACHE, THEN
17673                                   ;THE FIRST 512(10) LOCATIONS (BLOCK 0) WILL BE WRITTEN WITH ADDRESSES THAT
17674                                   ;WILL CAUSE A INCREMENTING PATTERN TO BE STORED IN THOSE LOCATIONS OF THE TAG
17675                                   ;STORE RAM. NEXT THE ENTIRE 4K ADDRESS RANGE WILL BE READ. THE HIT/MISS
17676                                   ;REGISTER SHOULD ONLY REPORT HITS ON THE ADDRESSES THAT WERE ALLOCATED. THIS
17677                                   ;PROCESS WILL BE REPEATED FOR EACH BLOCK.
17678                                   ;
17679                                   ;INITIALIZE LOW ADDRESS
17680                                   ;SETUP AND ENABLE MMU
17681                                   ;INITIALIZE BLOCK COUNTER
17682                                   ;DO UNTIL ALL BLOCKS TESTED (8 TIMES)
17683                                   ;.      FLUSH CACHE AND SET DIAGNOSTIC BIT
17684                                   ;.      LET CURRENT ADDRESS = LOW ADDRESS
17685                                   ;.      INITIALIZE ALLOCATION COUNTER
17686                                   ;.      DO UNTIL ENTIRE BLOCK ALLOCATED (1000 TIMES)
17687                                   ;.      .       READ CURRENT ADDRESS
17688                                   ;.      .       ELSE
17689                                   ;.      .       WRITE CURRENT ADDRESS
17690                                   ;.      .       ENDIF
17691                                   ;.      .       UPDATE CURRENT ADDRESS (ALSO ADD 200 TO PAR)
17692                                   ;.      .       FOR I/O PAGE WRITE THE SAME ADDRESS AS BEFORE
17693                                   ;.      ENDDO
17694                                   ;.      INITIALIZE GOOD ADDRESS
17695                                   ;.      INITIALIZE CURRENT ADDRESS
17696                                   ;.      INITIALIZE LOCATION COUNTER
17697                                   ;.      SAVE CONTENTS OF VECTOR 114
17698                                   ;.      LET VECTOR 114 POINT TO TAG STORE PARITY ABORT ROUTINE
17699                                   ;.      DO UNTIL ALL LOCATIONS CHECKED (1000 TIMES)
17700                                   ;.      .       INITIALIZE CHECK COUNTER
17701                                   ;.      .       DO UNTIL ADDRESS IN EACH BLOCK CHECKED
17702                                   ;.      .       .       READ CURRENT ADDRESS
17703                                   ;.      .       .       IF HIT THEN
17704                                   ;.      .       .       .       IF CURRENT ADDRESS NE GOOD ADDRESS THEN
17705                                   ;.      .       .       .       .       ERROR
17706                                   ;.      .       .       .       ENDIF
17707                                   ;.      .       .       ELSE
17708                                   ;.      .       .       .       IF CURRENT ADDRESS EQUAL GOOD ADDRESS THEN
17709                                   ;.      .       .       .       .       ERROR
17710                                   ;.      .       .       .       ENDIF
17711                                   ;.      .       .       ENDIF
17712                                   ;.      .       .       UPDATE GOOD ADDRESS (ADD #2)
17713                                   ;.      .       .       UPDATE CURRENT ADDRESS
17714                                   ;.      .       ENDDO
17715                                   ;.      .       UPDATE LOW ADDRESS (ADD #2000)
17716                                   ;.      ENDDO
17717                                   ;ENDDO
17718                                   ;DISABLE MMU
17719                                   ;RESTORE VECTOR 114
17720                                   ;ENDTST
17721
```

```
17722                                   ;;*****************************************************************
17723  106266  000004                   TST24:  SCOPE
17724  106270  013737  000004  001160           MOV     @#4,#TMP0               ;STORE TIMEOUT VECTOR
17725  106276  012737  107002  000004           MOV     #20$,@#4                ;POINT NEW
17726  106304  012737  140000  002766           MOV     #140000,LOWADD          ;INITIALIZE LOW ADDRESS (USE PAR6)
17727  106312  004737  131362                   JSR     PC,     INITMM          ;INITIALIZE MMU
17728  106316  005237  177572                   INC     SR0                     ;ENABLE MMU
17729  106322  012737  000020  172516           MOV     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
17730  106330  012737  177770  003114           MOV     #-10.,  LOOPIN          ;DO UNTIL ALL BLOCKS TESTED
17731  106336  012701  000000           1$:     MOV     #0,R1                   ;DO IN 2 WORDS TO AVOID PMI MEMORY
17732  106342  005037  172354           9$:     CLR     KIPAR6                  ;SET UP PAR6 FOR THIS TEST
17733  106346  012737  000402  177746           MOV     #402,   CCR             ;FLUSH CACHE AND SET DIAG BIT
17734  106354  013737  002766  110172           MOV     LOWADD, CURADD          ;GET FIRST ADDRESS IN CURRENT BLOCK
17735  106362  012737  177400  003112           MOV     #-400,  ALLCTR          ;DO UNTIL ALL ADDRESSES ALLOCATED
17736  106370  022737  002000  172354  2$:      CMP     #2000,  KIPAR6          ;IF ADDRESS LESS THAN 32K
17737  106376  002413                           BLT     3$                      ;THEN
17738  106400  052737  100000  172314           BIS     #BIT15,KIPDR6           ;SET BYPASS
17739  106406  017702  001560                   MOV     @CURADD,R2              ;STORE CURRENT DATA
17740  106412  042737  100000  172314           BIC     #BIT15,KIPDR6           ;ALLOCATE NEXT ACCESS
17741  106420  010277  001546                   MOV     R2,@CURADD              ;WRITE ALLOCATE
17742  106424  000402                           BR      4$                      ;ELSE
17743  106426  005077  001540           3$:     CLR     @CURADD                 ;WRITE CURRENT ADDRESS
17744  106432  062737  000002  110172   4$:     ADD     #2,     CURADD          ;UPDATE CURRENT ADDRESS
17745  106440  062737  000200  172354           ADD     #200,   KIPAR6
17746  106446  022737  177600  172354           CMP     #177600,KIPAR6          ;REACHED I/O PAGE?
17747  106454  001003                           BNE     10$                     ;BRANCH IF NOT
17748  106456  162737  000200  172354           SUB     #200,KIPAR6             ;DON'T UPDATE PAR FOR I/O PAGE
17749  106464  005237  003112           10$:    INC     ALLCTR                  ;IF ALL ADDRESSES ALLOCATED
17750  106470  002737                           BLT     2$                      ;ENDDO
17751  106472  052737  000004  177746           BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
17752  106500  013737  002766  002770           MOV     LOWADD, GOODAD          ;INITIALIZE GOOD ADDRESS
17753  106506  060137  002770                   ADD     R1,     GOODAD          ;
17754  106512  012737  140000  110172           MOV     #140000,CURADD          ;GET FIRST ADDRESS
17755  106520  060137  110172                   ADD     R1,     CURADD          ;START WITH 1 OR 2 LOCATION
17756  106524  005037  172354                   CLR     KIPAR6
17757  106530  072127  000006                   ASH     #6,R1                   ;
17758  106534  060137  172354                   ADD     R1,KIPAR6               ;MAKE SURE ON THE RIGHT BOUNDARY
17759  106540  012737  177600  003112           MOV     #-200.  ALLCTR          ;DO UNTIL ALL LOCATIONS CHECKED
17760  106546  012737  177770  110146   5$:     MOV     #-10.   DCOUNT          ;DO UNTIL ADDRESS IN EACH BLOCK CHECKED
17761  106554  013737  110172  001122   6$:     MOV     CURADD, @#BDADR         ;IN CASE OF ERRORS
17762  106562  042737  160000  001122           BIC     #160000,@#BDADR         ;CLEAR PAR BITS
17763  106570  042737  000004  177746           BIC     #BIT02, CCR             ;CLEAR FORCE MISS
17764  106576  005777  001370                   TST     @CURADD                 ;READ CURRENT ADDRESS
17765  106602  013737  177752  110152           MOV     HITMIS, RECDAT          ;STORE REGISTER
17766  106610  032737  000004  110152           BIT     #BIT2,  RECDAT          ;IF ACCESS WAS A HIT
17767  106616  001406                           BEQ     7$                      ;THEN
17768  106620  023737  110172  002770           CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS NOT EQUAL TO GOOD
17769  106626  001407                           BEQ     8$                      ;ADDRESS THEN
17770  106630  104046                           ERROR   .46                     ;ERROR IN TAG STORE
17771  106632  000405                           BR      8$                      ;
17772  106634  023737  110172  002770   7$:     CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS EQUAL GOOD ADDRESS
17773  106642  001001                           BNE     8$                      ;THEN
17774  106644  104047                           ERROR   .47                     ;ERROR IN TAG STORE
17775  106646  062737  001000  110172   8$:     ADD     #1000,  CURADD          ;UPDATE TO NEXT BLOCK
17776  106654  052737  000004  177746           BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
17777  106662  005237  110146                   INC     DCOUNT                  ;IF ADDRESS NOT CHECKED FOR EACH BLOCK
```

```
 17778  106666  002732                        BLT     6$                      ;ENDDO
 17779  106670  062737  000004  002770        ADD     #4,     GOODAD          ;UPDATE GOOD ADDRESS
 17780  106676  162737  007774  110172        SUB     #7774,  CURADD          ;INCREMENT IN 2 WORDS
 17781  106704  062737  000400  172354        ADD     #400,   KIPAR6
 17782  106712  005237  003112                INC     ALLCTR                  ;IF ALL ADDRESSES CHECKED
 17783  106716  002713                        BLT     5$                      ;ENDDO
 17784  106720  062701  000002                ADD     #2,     R1              ;PREPARE FOR THE 2ND PASS THRU
 17785  106724  022701  000002                CMP     #2,     R1              ;DONE TWICE?
 17786  106730  001604                        BEQ     9$
 17787  106732  062737  002000  002766        ADD     #2000,  LOWADD          ;UPDATE TO NEXT BLOCK TO BE ALLOCATED
 17788  106740  005237  003114                INC     LOOPIN                  ;IF ALL BLOCKS WERE TESTED
 17789  106744  002002                        BGE     15$                     ;
 17790  106746  000137  106336                JMP     1$                      ;ENDDO
 17791  106752  012737  000400  177746  15$:  MOV     #400,   CCR             ;CLEAR DIAGNOSTIC BIT
 17792  106760  005037  177572                CLR     SR0                     ;DISABLE MMU
 17793  106764  013737  001160  000004        MOV     $TMP0,@#4               ;RESTORE TIMEOUT VECTOR
 17794  106772  042737  000020  172516        BIC     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
 17795  107000  000401                        BR      TST25           ;;GO TO NEXT TEST
 17796
 17797  107002  000002                20$:    RTI
 17798
 17799
```

```
 17800                                          .SBTTL  TEST - STANDALONE MODE TEST
 17801                                          ;THIS TEST VERIFIES THAT NMX CAN BE CREATED IN STANDALONE MODE.
 17802                                          ;
 17803                                          ;ALLOCATE INSTRUCTIONS IN CACHE
 17804                                          ;GUARANTY MISS ON TEST LOCATION
 17805                                          ;IN STANALONE MODE ACCESS TEST LOCATION
 17806                                          ;IF NO TIMEOUT THEN
 17807                                          ;.      ERROR
 17808                                          ;ENDIF
 17809                                          ;
 17810                                          ;;*************************/*************************************
 17811  107004  000004                         TST25:  SCOPE
 17812  107006  012737  000400  177746                 MOV     #400,CCR                ;FLUSH CACHE
 17813  107014  012701  107014                         MOV     #.,R1                   ;START WITH CURRENT INSTRUCTION
 17814  107020  005721                         1$:     TST     (R1)+                   ;READ A WORD
 17815  107022  022701  107104                         CMP     #10$,R1                 ;DONE?
 17816  107026  001374                                 BNE     1$                      ;IF NOT, CONTINUE
 17817  107030  005737  020000                         TST     @#20000                 ;TO GUARANTY MISS ON 0
 17818  107034  013702  000004                         MOV     @#4,R2                  ;STORE TIMEOUT VECTOR
 17819  107040  012737  107066  000004                 MOV     #5$,@#4                 ;POINT NEW TO THE TEST
 17820  107046  012737  000340  000006                 MOV     #340,@#6                ;AT PRIORITY 7
 17821  107054  052737  000400  177520                 BIS     #BIT08,BCSR             ;GO TO STANDALONE MODE
 17822  107062  005737  000000                         TST     @#0                     ;MISS, SHOULD TIMEOUT
 17823  107066  042737  000400  177520  5$:     BIC     #BIT08,BCSR             ;CLEAR STANDALONE BIT
 17824  107074  022716  107066                         CMP     #5$,(SP)                ;TIMEOUT?
 17825  107100  001401                                 BEQ     10$                     ;IF YES, BRANCH
 17826  107102  104044                                 ERROR   .44
 17827  107104  012706  001100          10$:    MOV     #1100,SP                ;RESTORE STACK
 17828  107110  010237  000004                         MOV     R2,@#4                  ;AND TIMEOUT VECTOR
```

```
17829                                   .SBTTL  TEST  - MOVING INVERSIONS TEST FOR DATA RAMS
17830                                   ;MOVING INVERSIONS TEST FOR DATA RAMS - THE TEST IS STARTED AFTER LOADING THE
17831                                   ;RAM STORE WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A
17832                                   ;1 IS SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE
17833                                   ;ADDRESS IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF
17834                                   ;THE WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED
17835                                   ;PLUGGING IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESSING IN THE DOWNWARD
17836                                   ;DIRECTION. FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE
17837                                   ;LSB. TO SAVE TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING
17838                                   ;ONLY A SINGLE BIT AT A TIME EVERY FOURTH BIT WILL BE DONE CONCURRENTLY.
17839                                   ;THIS TEST RUNS IN STANDALONE MODE.
17840                                   ;
17841                                   ;BGNTST
17842                                   ;SETUP AND ENABLE MMU
17843                                   ;SETUP CCR TO ABORT PARITY ERRORS
17844                                   ;CLEAR CACHE
17845                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
17846                                   ;LET FWDSEQ = #1
17847                                   ;LET ADDLSB = #1
17848                                   ;DO UNTIL ADDLSB EQ #20000
17849                                   ;.      LET CURDAT = 0
17850                                   ;.      LET RITEDA = #1
17851                                   ;.      LET NEWDAT = #1
17852                                   ;.      IF FWDSEQ = #1 THEN
17853                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
17854                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
17855                                   ;.      ELSE
17856                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
17857                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
17858                                   ;.      ENDIF
17859                                   ;.      LET CURADD = FSTADD
17860                                   ;.      LET DCOUNT = #C
17861                                   ;.      SAVE CONTENTS OF VECTOR 114
17862                                   ;.      LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
17863                                   ;.      DO UNTIL DCOUNT EQ #10
17864                                   ;.      .       LET RECDAT = @CURADD
17865                                   ;.      .       IF RECDAT NE CURDAT THEN
17866                                   ;.      .       .       LET R1 EQUAL CURRENT DATA
17867                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
17868                                   ;.      .       ENDIF
17869                                   ;.      .       IF DCOUNT GT #3 THEN
17870                                   ;.      .       .       LET @CURADD = @CURADD CLEARBY RITEDA
17871                                   ;.      .       ELSE
17872                                   ;.      .       .       LET @CURADD = @CURADD SETBY RITEDA
17873                                   ;.      .       ENDIF
17874                                   ;.      .       LET RECDAT = @CURADD
17875                                   ;.      .       IF RECDAT NE NEWDAT THEN
17876                                   ;.      .       .       LET R1 EQUAL NEW DATA
17877                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
17878                                   ;.      .       ENDIF
17879                                   ;.      .       IF CURADD EQ LASTAD THEN
17880                                   ;.      .       .       IF RITEDA = #210 THEN
17881                                   ;.      .       .       .       IF DCOUNT NE #7 THEN
17882                                   ;.      .       .       .       .       LET CURDAT = #377
17883                                   ;.      .       .       .       .       LET RITEDA = #1
17884                                   ;.      .       .       .       .       LET NEWDAT = #376
```

```
17885                                   ;.      .       .       .       ENDIF
17886                                   ;.      .       .       ELSE
17887                                   ;.      .       .       .       LET CURDAT = NEWDAT
17888                                   ;.      .       .       .       ROTATE RITEDA
17889                                   ;.      .       .       .       IF DCOUNT GT #3 THEN
17890                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT CLEAREDBY RITEDA
17891                                   ;.      .       .       .       ELSE
17892                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA
17893                                   ;.      .       .       .       ENDIF
17894                                   ;.      .       .       ENDIF
17895                                   ;.      .       .       INCREMENT DCOUNT
17896                                   ;.      .       .       LET CURADD = FSTADD
17897                                   ;.      .       ELSE
17898                                   ;.      .       .       IF FWDSEQ = #1 THEN
17899                                   ;.      .       .       .       LET CURADD = CURADD + ADDLSB
17900                                   ;.      .       .       .       IF CARRY THEN
17901                                   ;.      .       .       .       .       LET CURADD = CURADD + #1
17902                                   ;.      .       .       .       ENDIF
17903                                   ;.      .       .       ELSE
17904                                   ;.      .       .       .       LET CURADD = CURADD - ADDLSB
17905                                   ;.      .       .       .       IF CARRY THEN
17906                                   ;.      .       .       .       .       LET CURADD = LASTAD - ADDLSB
17907                                   ;.      .       .       .       .       LET CURADD = CURADD - #1
17908                                   ;.      .       .       .       ENDIF
17909                                   ;.      .       .       ENDIF
17910                                   ;.      .       ENDIF
17911                                   ;.      ENDDO
17912                                   ;.      IF FWDSEQ EQ #1 THEN
17913                                   ;.      .       LET FWDSEQ = #0
17914                                   ;.      ELSE
17915                                   ;.      .       ROTATE ADDLSB
17916                                   ;.      .       LET FWDSEQ = #1
17917                                   ;.      ENDIF
17918                                   ;ENDDO
17919                                   ;RESTORE VECTOR 114
17920                                   ;ENDTST
17921
17922                                   ;;*****************************************************************
17923   107114  000004                  TST26:  SCOPE
17924   107116  032777  000200  072014          BIT     #BIT07,@SWR             ;RUN THIS TEST?
17925   107124  001002                          BNE     100$                    ;IF SET, GO DO IT
17926   107126  000137  110174                  JMP     ENDMOV                  ;OTHERWISE, GO TO NEXT TEST
17927   107132  004737  131362          100$:   JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
17928   107136  012737  002000  172354          MOV     #2000,KIPAR6            ;START ON 32K BOUNDARY
17929   107144  005237  177572                  INC     SR0                     ;TURN ON MMU
17930   107150  052737  000002  177746          BIS     #2,     CCR             ;SET DIAG. BIT
17931   107156  013737  000004  001160          MOV     @#4,    @TMP0           ;SAVE 4
17932                                   ;
17933                                   ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
17934                                   ;
17935   107164  012703  140000                  MOV     #140000,R3              ;START FOR THE TEST
17936   107170  012704  150000                  MOV     #150000,R4              ;LOWER BOUNDARY TEST AREA
17937   107174  012705  157777                  MOV     #157777,R5              ;HIGH BOUNDARY TEST AREA
17938   107200  000406                          BR      2$
17939   107202  012703  150000          1$:     MOV     #150000,R3              ;START OF THE TEST
17940   107206  012704  140000                  MOV     #140000,R4              ;LOWER BOUNDARY TEST AREA
```

```
17941  107212  012705  147777                  MOV     #147777,R5              ;HIGH BOUNDARY
17942  107216  012702  107330          2$:     MOV     #STMOVI,R2              ;START WITH CURRENT
17943  107222  010300                          MOV     R3,R0                   ;MOVE TO UPPER 4K
17944  107224  012220                  3$:     MOV     (R2)+,(R0)+             ;WORD BY WORD
17945  107226  022702  110174                  CMP     #ENDMOV,R2              ;ALL DONE
17946  107232  001374                          BNE     3$                      ;
17947  107234  010400                          MOV     R4,R0                   ;CLEAR CACHE UNDER TEST
17948  107236  012701  004000                  MOV     #4000,  R1
17949  107242  005020                  4$:     CLR     (R0)+
17950  107244  077102                          SOB     R1,     4$
17951  107246  004713                          JSR     PC,(R3)                 ;GO DO THE ROUTINE
17952  107250  005702                          TST     R2                      ;ANY ERRORS?
17953  107252  001411                          BEQ     5$                      ;IF NOT, CONTINUE
17954  107254  010037  110152                  MOV     R0,RECDAT               ;DATA RECEIVED
17955  107260  010237  001122                  MOV     R2,$BDADR               ;ADDRESS RECIEVED
17956  107264  042737  140000  001122          BIC     #140000,$BDADR          ;STRIP OF PAR BITS
17957  107272  104043                          ERROR   +43                     ;
17958  107274  000403                          BR      6$                      ;EXIT
17959  107276  022703  150000          5$:     CMP     #150000,R3              ;DONE FOR BOTH HALVES?
17960  107302  001337                          BNE     1$                      ;IF NOT, DO AGAIN
17961  107304  013737  001160  000004  6$:     MOV     $TMP0,@#4               ;RESTORE TIMEOUT VECTOR
17962  107312  012737  000400  177746          MOV     #400,CCR                ;INIT CCR FOR EXIT
17963  107320  005037  177572                  CLR     SR0                     ;TURN OFF MMU
17964  107324  000137  110174                  JMP     ENDMOV                  ;GO TO NEXT TEST
17965
17966                                  .DSABL AMA
17967  107330  052737  000400  177520  STMOVI: BIS     #BIT08,@#CSR            ;STANDALONE MODE
17968  107336  005002                          CLR     R2                      ;ERROR INDICATOR
17969  107340  012767  000001  000606          MOV     #1,     FWDSEQ          ;INIT UPWARD ADDRESSING INDICATOR
17970  107346  012767  000001  000602          MOV     #1,     ADDLSB          ;INIT LSB AND DO LOOP UNTIL SHIFTED OUT
17971  107354  042737  100000  172300          BIC     #100000,@#KIPDR0        ;NO BYPASS
17972  107362  012737  172360  000004          MOV     #KDPAR0,@#4             ;ALLOCATE TIMEOUT VECTOR
17973  107370  012737  000340  000006          MOV     #340,   @#6             ;AT PRIORITY 7
17974  107376  012737  000006  172360          MOV     #6,     @#KDPAR0        ;PUT RETURN
17975  107404  052737  100000  172300          BIS     #100000,@#KIPDR0        ;BYPASS
17976  107412  005067  000546          TSTLUP: CLR     CURDAT                  ;INIT CURRENT DATA
17977  107416  012767  000021  000534          MOV     #21.    RITEDA          ;INIT DATA TO BE WRITTEN
17978  107424  012767  000021  000530          MOV     #21.    NEWDAT          ;INIT EXPECTED DATA
17979  107432  005767  000516                  TST     FWDSEQ                  ;IF ADDRESSING UPWARD
17980  107436  001405                          BEQ     1$                      ;THEN
17981  107440  010467  000522                  MOV     R4,FSTADD               ;LET FIRST ADDRESS EQUAL LOWEST VALUE
17982  107444  010567  000520                  MOV     R5,LSTADD               ;LET LAST ADDRESS EQUAL HIGHEST VALUE
17983  107450  000404                          BR      2$                      ;ELSE
17984  107452  010567  000510          1$:     MOV     R5,FSTADD               ;LET FIRST ADDRESS EQUAL HIGHEST VALUE
17985  107456  010467  000506                  MOV     R4,LSTADD               ;LET LAST ADDRESS EQUAL LOWEST VALUE
17986  107462  016767  000500  000502  2$:     MOV     FSTADD, CURADD          ;LET CURRENT ADDRESS EQUAL FIRST ADDRESS
17987  107470  005067  000452                  CLR     DCOUNT                  ;INIT LOOP COUNTER
17988  107474  022767  000010  000444  BGNTLP: CMP     #10,    DCOUNT          ;DO LOOP 8 TIMES (4 TIMES TO WRITE 1'S
17989  107502  001567                          BEQ     ENDTLP                  ;AND 4 TIMES TO WRITE BACK 0'S)
17990                                          ;
17991                                  ; DON'T REWRITE TIMEOUT VECTOR
17992                                          ;
17993  107504  016700  000462                  MOV     CURADD, R0              ;STORE CURRENT ADDRESS
17994  107510  042700  170000                  BIC     #170000,R0              ;LEAVE ONLY LOW 4K BITS
17995  107514  022700  000004                  CMP     #4,     R0              ;TIMOUT VECTOR?
17996  107520  001531                          BEQ     8$                      ;IF SO, DON'T REWRITE IT
```

```
 17997  107522  022700  000005                  CMP     #5.     R0              ;
 17998  107526  001526                          BEQ     8$                      ;
 17999  107530  022700  000006                  CMP     #6.     R0              ;
 18000  107534  001523                          BEQ     8$                      ;
 18001  107536  022700  000007                  CMP     #7.     R0              ;
 18002  107542  001520                          BEQ     8$                      ;
 18003                                  ;
 18004                                  ;CHECK PATTERNS NOW
 18005                                  ;
 18006  107544  016701  000414                  MOV     CURDAT, R1              ;MOVE GOOD DATA TO R1
 18007  107550  117767  000416  000374          MOVB    @CURADD,RECDAT          ;FIRST READ LOCATION
 18008  107556  126767  000370  000400          CMPB    RECDAT, CURDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
 18009  107564  001402                          BEQ     1$                      ;DATA THEN
 18010  107566  000167  000334                  JMP     EXBAD                   ;EXIT TEST
 18011  107572  016701  000364          1$:     MOV     NEWDAT, R1              ;MOVE GOOD DATA TO R1
 18012  107576  022767  000003  000342          CMP     #3.     DCOUNT          ;IF LOOP COUNT GREATER THAN 3
 18013  107604  002004                          BGE     2$                      ;THEN
 18014  107606  146777  000346  000356          BICB    RITEDA, @CURADD         ;CLEAR TEST DATA BY WRITE DATA
 18015  107614  000403                          BR      3$                      ;ELSE
 18016  107616  156777  000336  000346  2$:     BISB    RITEDA, @CURADD         ;SET TEST DATA BY WRITE DATA
 18017  107624  117767  000342  000320  3$:     MOVB    @CURADD,RECDAT          ;DO READ AFTER WRITE
 18018  107632  126767  000314  000322          CMPB    RECDAT, NEWDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
 18019  107640  001402                          BEQ     4$                      ;DATA THEN
 18020  107642  000167  000260                  JMP     EXBAD                   ;EXIT TEST
 18021  107646  026767  000320  000314  4$:     CMP     CURADD, LSTADD          ;IF CURRENT ADDRESS EQUALS LAST ADDRESS
 18022  107654  001053                          BNE     8$                      ;THEN
 18023  107656  022767  000210  000274          CMP     #210,   RITEDA          ;AND IF WRITE PATTERN EQUALS LAST
 18024  107664  001016                          BNE     5$                      ;PATTERN THEN
 18025  107666  022767  000007  000252          CMP     #7.     DCOUNT          ;AND TEST IS NOT ON LAST LOOP
 18026  107674  001435                          BEQ     7$                      ;THEN
 18027  107676  012767  000377  000260          MOV     #377,   CURDAT          ;SET UP TO WRITE 0'S
 18028  107704  012767  000021  000246          MOV     #21,    RITEDA
 18029  107712  012767  000356  000242          MOV     #356,   NEWDAT
 18030  107720  000423                          BR      7$                      ;ELSE
 18031  107722  016767  000234  000234  5$:     MOV     NEWDAT, CURDAT          ;SET UP FOR NEXT DATA PATTERN
 18032  107730  000241                          CLC
 18033  107732  106167  000222                  ROLB    RITEDA
 18034  107736  005567  000216                  ADC     RITEDA                  ;GET A PATTERN
 18035  107742  022767  000003  000176          CMP     #3.     DCOUNT          ;AND IF DOWNWARD ADDRESSING
 18036  107750  002004                          BGE     6$                      ;THEN
 18037  107752  046767  000202  000202          BIC     RITEDA, NEWDAT          ;LET NEWDAT BE CLEARED BY WRITE DATA
 18038  107760  000403                          BR      7$                      ;ELSE
 18039  107762  056767  000172  000172  6$:     BIS     RITEDA, NEWDAT          ;LET NEWDAT BE SET BY WRITE DATA
 18040  107770  005267  000152          7$:     INC     DCOUNT                  ;UPDATE LOOP COUNTER
 18041  107774  016767  000166  000170          MOV     FSTADD, CURADD          ;REINIT FIRST ADDRESS FOR THIS PASS
 18042  110002  000426                          BR      10$                     ;ELSE
 18043  110004  005767  000144          8$:     TST     FWDSEQ                  ;IF ADDRESSING UPWARD
 18044  110010  001412                          BEQ     9$                      ;THEN
 18045  110012  066767  000140  000152          ADD     ADDLSB, CURADD          ;CALCULATE NEXT HIGHER ADDRESS
 18046  110020  020567  000146                  CMP     R5,CURADD               ;IF CURRENT ADDRESS HAS BEEN INCREASED
 18047  110024  002015                          BGE     10$                     ;ABOVE HIGHEST ADDRESS THEN
 18048  110026  162767  007777  000136          SUB     #7777,  CURADD          ;ROLL ADDRESS BACK
 18049  110034  000411                          BR      10$                     ;ELSE
 18050  110036  166767  000114  000126  9$:     SUB     ADDLSB, CURADD          ;CALCULATE NEXT LOWER ADDRESS
 18051  110044  020467  000122                  CMP     R4,CURADD               ;IF CURRENT ADDRESS HAS BEEN DECREASED
 18052  110050  003403                          BLE     10$                     ;BELOW LOWEST ADDRESS THEN
```

```
18053  110052  062767  007777  000112          ADD     #7777,  CURADD          ;ROLL ADDRESS BACK
18054  110060  000605                  10$:     BR      BGNTLP                  ;ENDDO
18055  110062  005767  000066          ENDTLP:  TST     FWDSEQ                  ;IF ADDRESSING UPWARD FINISHED
18056  110066  001404                          BEQ     1$                      ;THEN
18057  110070  005067  000060                  CLR     FWDSEQ                  ;DO ADDRESSING DOWNWARD
18058  110074  000167  177312                  JMP     TSTLUP
18059  110100  012767  000001  000046  1$:     MOV     #1,     FWDSEQ          ;SET ADDRESSING UPWARD INDICATOR
18060  110106  006167  000044                  ROL     ADDLSB                  ;UPDATE LSB TO NEXT POSITION
18061  110112  022767  020000  000036  ENDLUP: CMP     #20000,ADDLSB           ;ALL DONE?
18062  110120  001406                          BEQ     EXITST                  ;ENDDO
18063  110122  000167  177264                  JMP     TSTLUP
18064  110126  016700  000020          EXBAD:   MOV     RECDAT,R0               ;STORE RECEIVED DATA
18065  110132  016702  000034                  MOV     CURADD,R2               ;STORE ADDRESS
18066  110136  042737  000400  177520  EXITST: BIC     #BIT08,@#8CSR           ;OUT OF STANDALONE
18067  110144  000207                          RTS     PC
18068                                  .ENABL AMA
18069
18070  110146  000000                  DCOUNT: .WORD   0
18071  110150  000000                  EXPDAT: .WORD   0               ;STORES EXPECTED (GOOD) DATA FOR COMPARISONS
18072  110152  000000                  RECDAT: .WORD   0               ;STORES RECIEVED DATA TO BE VERIFIED
18073  110154  000000                  FWDSEQ: .WORD   0               ;USED TO INDICATE DIRECTION OF ADDRESSING
18074  110156  000000                  ADDLSB: .WORD   0               ;STORES LEAST SIGNIFICANT BIT FOR RAM TESTS
18075  110160  000000                  RITEDA: .WORD   0               ;STORES WRITE DATA FOR RAM TESTS
18076  110162  000000                  NEWDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18077  110164  000000                  CURDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18078  110166  000000                  FSTADD: .WORD   0               ;STORES FIRST ADDRESS IN ADDRESSING SEQUENCE
18079  110170  000000                  LSTADD: .WORD   0               ;STORES LAST ADDRESS IN ADDRESSING SEQUENCE
18080  110172  000000                  CURADD: .WORD   0               ;STORES CURRENT ADDRESS FOR RAM TESTS
18081
18082  110174                          ENDMOV:
```

```
18083                                   .SBTTL  TEST  - MOVING INVERSIONS TEST FOR TAG STORE
18084                                   ;MOVING INVERSIONS TEST - THE TEST IS STARTED AFTER LOADING THE RAM STORE
18085                                   ;WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A 1 IS
18086                                   ;SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE ADDRESS
18087                                   ;IS READ  0 VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF THE
18088                                   ;WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED PLUGGING
18089                                   ;IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESS IN THE DOWARD DIRECTION.
18090                                   ;FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE LSB. TO SAVE
18091                                   ;TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING ONLY A SINGLE
18092                                   ;BIT AT A TIME EVERY THIRD BIT WILL BE DONE CONCURRENTLY. ALSO NOTE THAT
18093                                   ;SINCE THE TAG STORE CANNOT BE DIRECTLY ACCESSED THE ENTIRE PATTERN MUST BE
18094                                   ;DONE BY DOING MEMORY CYCLES TO THE CORRECT BUS ADDRESSES. TO DO THIS THE
18095                                   ;TEST WILL BE DONE .ITH THE DIAGNOSTIC BIT IN THE CCR (BIT 1) SET TO A 1.
18096                                   ;THIS TEST RUNS IN STANDALONE MODE.
18097                                   ;
18098                                   ;
18099                                   ;BGNTST
18100                                   ;SETUP AND ENABLE MMU
18101                                   ;SETUP CCR TO ABORT PARITY ERRORS
18102                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
18103                                   ;LET FWDSEQ = #1
18104                                   ;LET ADDLSB = #1
18105                                   ;DO UNTIL ADDLSB EQ #20000
18106                                   ;.      LET NEWDAT = 22200
18107                                   ;.      LET CURDAT = 0
18108                                   ;.      IF FWDSEQ = #1 THEN
18109                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18110                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18111                                   ;.      ELSE
18112                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18113                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18114                                   ;.      ENDIF
18115                                   ;.      LET DCOUNT = #0
18116                                   ;.      DO UNTIL DCOUNT EQ #10
18117                                   ;.      .       READ CURADD USING CURDAT AS PAR
18118                                   ;.      .       IF MISS THEN
18119                                   ;.      .       .       ERROR
18120                                   ;.      .       ENDIF
18121                                   ;.      .       WRITE CURADD USING NEWDAT AS PAR
18122                                   ;.      .       ENDIF
18123                                   ;.      .       IF HIT THEN
18124                                   ;.      .       .       ERROR
18125                                   ;.      .       ENDIF
18126                                   ;.      .       READ CURADD USING NEWDAT AS PAR
18127                                   ;.      .       IF MISS THEN
18128                                   ;.      .       .       ERROR
18129                                   ;.      .       ENDIF
18130                                   ;.      .       IF CURADD EQ LASTAD THEN
18131                                   ;.      .       .       LET CURDAT = NEWDAT
18132                                   ;.      .       .       IF DCOUNT < #1 THEN
18133                                   ;.      .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA ROTATED LEFT
18134                                   ;.      .       .       ENDIF
18135                                   ;.      .       .       IF DCOUNT > #1 THEN
18136                                   ;.      .       .       .       LET NEWDAT = NEWDAT CLR BY RITEDA ROTATED LEFT
18137                                   ;.      .       .       ELSE
18138                                   ;.      .       .       .       LET NEWDAT = #155400 (NO ALL 1'S)
```

```
18139                                 ;.      .       .       .       LET RITEDA = #22200
18140                                 ;.      .       .       ENDIF
18141                                 ;.      .       .       INCREMENT DCOUNT
18142                                 ;.      .       ELSE
18143                                 ;.      .       .       IF FWDSEQ = #1 THEN
18144                                 ;.      .       .       .       LET CURADD = CURADD + ADDLSB
18145                                 ;.      .       .       .       IF CURADD GE HIGH ADDRESS THEN
18146                                 ;.      .       .       .       .       LET CURADD = CURADD + #7776
18147                                 ;.      .       .       .       ENDIF
18148                                 ;.      .       .       ELSE
18149                                 ;.      .       .       .       LET CURADD = CURADD - ADDLSB
18150                                 ;.      .       .       .       IF CURADD LE LOW ADDRESS THEN
18151                                 ;.      .       .       .       .       LET CURADD = CURADD - #7776
18152                                 ;.      .       .       .       ENDIF
18153                                 ;.      .       .       ENDIF
18154                                 ;.      .       ENDIF
18155                                 ;.      ENDDO
18156                                 ;.      IF FWDSEQ EQ #1 THEN
18157                                 ;.      .       LET FWDSEQ = #0
18158                                 ;.      ELSE
18159                                 ;.      .       ROTATE ADDLSB
18160                                 ;.      .       LET FWDSEQ = #1
18161                                 ;.      ENDIF
18162                                 ;ENDDO
18163                                 ;RESTORE PARITY ABORT VECTOR
18164                                 ;ENDTST
18165
18166                                 ;;*****************************************************************
18167  110174  000004                 TST27:  SCOPE
18168  110176  032777  000100  070734         BIT     #BIT06,@SWR             ;RUN THIS TEST?
18169  110204  001002                         BNE     100$                    ;IF SET, GO DO IT
18170  110206  000137  111262                 JMP     ENDTAG                  ;OTHERWISE, GO TO NEXT TEST
18171  110212  004737  131362         100$:   JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
18172  110216  005237  177572                 INC     SR0                     ;TURN ON MMU
18173  110222  012737  000020  172516         MOV     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
18174  110230  052737  000002  177746         BIS     #2,     CCR             ;SET DIAG. BIT
18175  110236  013737  000004  001160         MOV     #4,     $TMPO           ;STORE TIMEOUT VECTOR
18176                                 ;
18177                                 ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
18178                                 ;
18179  110244  012703  140000                 MOV     #140000,R3              ;START FOR THE TEST
18180  110250  012704  130000                 MOV     #130000,R4              ;LOWER BOUNDARY TEST AREA
18181  110254  012705  137776                 MOV     #137776,R5              ;HIGH BOUNDARY TEST AREA
18182  110260  000406                         BR      2$
18183  110262  012703  150000         1$:     MOV     #150000,R3              ;START OF THE TEST
18184  110266  012704  120000                 MOV     #120000,R4              ;LOWER BOUNDARY TEST AREA
18185  110272  012705  127776                 MOV     #127776,R5              ;HIGH BOUNDARY
18186  110276  012737  002000  172354  2$:     MOV     #2000,KIPAR6            ;ABOVE 32K
18187  110304  012702  110136                 MOV     #EXITST,R2              ;START WITH CURRENT
18188  110310  010300                         MOV     R3,R0                   ;MOVE TO UPPER 4K
18189  110312  012220                 3$:     MOV     (R2)+,(R0)+             ;WORD BY WORD
18190  110314  022702  111262                 CMP     #ENDTAG,R2              ;ALL DONE?
18191  110320  001374                         BNE     3$                      ;
18192  110322  012700  110416                 MOV     #STMOVT,R0              ;ADDRESS OF ROUTINE
18193  110326  162700  110136                 SUB     #EXITST,R0              ;PROPER OFSET
18194  110332  050003                         BIS     R0,R3                   ;
```

```
18195  110334  004713                          JSR     PC,(R3)                 ;GO DO THE ROUTINE
18196  110336  005700                          TST     R0                      ;ANY ERRORS?
18197  110340  001407                          BEQ     4$                      ;CONTINUE
18198  110342  010037  001122                  MOV     R0,$BDADR               ;STORE FAILED ADDRESS
18199  110346  042737  160000  001122          BIC     #160000,$BDADR          ;CLEAR PAR
18200  110354  104050                          ERROR   ^D50
18201  110356  000403                          BR      5$                      ;EXIT TEST
18202  110360  022703  150000          4$:     CMP     #150000,R3              ;DONE FOR BOTH HALVES?
18203  110364  103736                          BLO     1$                      ;IF NOT, DO AGAIN
18204  110366  013737  001160  000004  5$:     MOV     $TMP0,  @#4             ;RESTORE TIMEOUT VECTOR
18205  110374  012737  000400  177746          MOV     #400,   CCR             ;INIT CCR FOR EXIT
18206  110402  005037  177572                  CLR     SR0                     ;TURN OFF MMU
18207  110406  005037  172516                  CLR     MMR3                    ;
18208  110412  000137  111262                  JMP     ENDTAG                  ;EXIT TEST
18209
18210                                  .DSABL AMA
18211  110416  052737  000400  177520 STMOVT:  BIS     #BIT08,@#BCSR           ;STANDALONE MODE
18212  110424  042737  100000  172312          BIC     #BIT15, @#KIPDR5        ;NO BYPASS
18213  110432  042737  100000  172300          BIC     #100000,@#KIPDR0        ;NO BYPASS
18214  110440  012737  172360  000004          MOV     #KDPAR0,@#4             ;ALLOCATE TIMEOUT VECTOR
18215  110446  012737  000340  000006          MOV     #340,   @#6             ;AT 7
18216  110454  012737  000006  172360          MOV     #6,     @#KDPAR0        ;PUT RETURN
18217  110462  052737  100000  172300          BIS     #100000,@#KIPDR0        ;BYPASS
18218  110470  005037  172352                  CLR     @#KIPAR5                ;
18219  110474  010400                          MOV     R4,R0                   ;CLEAR CACHE UNDER TEST
18220  110476  012701  004000                  MOV     #4000,  R1              ;
18221  110502  005020          4$:     CLR     (R0)+                           ;
18222  110504  077102                          SOB     R1,     4$              ;
18223  110506  005000                          CLR     R0                      ;CLEAR ERROR FLAG
18224  110510  012767  000001  177436          MOV     #1,     FWDSEQ          ;SET UPWARD ADDRESSING INDICATOR
18225  110516  012767  000002  177432          MOV     #2,     ADDLSB          ;INITIALIZE LEAST SIGNIFIGANT BIT
18226  110524  005067  177434  TSLOOP: CLR     CURDAT                          ;OLD DATA
18227  110530  012767  022200  177424          MOV     #22200,NEWDAT           ;SET ADDRESS BITS
18228  110536  012767  022200  177414          MOV     #22200,RITEDA           ;SET ADDRESS BITS
18229  110544  005767  177404                  TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18230  110550  001405                          BEQ     1$                      ;THEN
18231  110552  010467  177410                  MOV     R4,     FSTADD          ;FIRST ADDRESS WILL BE LOWEST ADDRESS
18232  110556  010567  177406                  MOV     R5,     LSTADD          ;AND LAST ADDRESS WILL BE HIGHEST
18233  110562  000404                          BR      2$                      ;ELSE
18234  110564  010567  177376          1$:     MOV     R5,     FSTADD          ;FIRST ADDRESS WILL BE HIGHEST ADDRESS
18235  110570  010467  177374                  MOV     R4,     LSTADD          ;AND LAST ADDRESS WILL BE LOWEST
18236  110574  005067  177346          2$:     CLR     DCOUNT                  ;INITIALIZE LOOP COUNTER
18237  110600  016767  177362  177364          MOV     FSTADD, CURADD
18238                                  ;
18239                                  ; DON'T REWRITE TIMEOUT VECTOR
18240                                  ;
18241  110606  016700  177360          3$:     MOV     CURADD, R0              ;STORE CURRENT ADDRESS
18242  110612  042700  170000                  BIC     #170000,R0              ;LEAVE ONLY LOW 4K BITS
18243  110616  022700  000004                  CMP     #4,     R0              ;TIMOUT VECTOR?
18244  110622  001526                          BEQ     16$                     ;IF SO, DON'T REWRITE IT
18245  110624  022700  000006                  CMP     #6,     R0              ;OR PRIORITY
18246  110630  001523                          BEQ     16$                     ;
18247  110632  016737  177326  172352          MOV     CURDAT, @#KIPAR5        ;GET OLD PATTERN
18248  110640  005777  177326                  TST     @CURADD                 ;OLD DATA OK?
18249  110644  013767  177752  177300          MOV     @#HITMIS,RECDAT
18250  110652  032767  000004  177272  5$:     BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
```

```
18251  110660  001002                          BNE     6$                      ;THEN
18252  110662  000167  000346                  JMP     EXBAD2                  ;ERROR, EXIT
18253  110666  016737  177270  172352  6$:     MOV     NEWDAT, @#KIPAR5        ;GET NEW PATTERN
18254  110674  005077  177272                  CLR     @CURADD                 ;REGISTER AND WRITE LOCATION
18255  110700  013767  177752  177244          MOV     @#HITMIS,RECDAT
18256  110706  032767  000004  177236  8$:     BIT     #BIT02, RECDAT          ;IF ACCESS WAS A HIT
18257  110714  001406                          BEQ     9$                      ;THEN
18258  110716  032767  000010  177226  10$:    BIT     #BIT03, RECDAT          ;MISS?
18259  110724  001402                          BEQ     9$                      ;IF SO, CONTINUE
18260  110726  000167  000302                  JMP     EXBAD2                  ;ERROR
18261  110732  005777  177234          9$:     TST     @CURADD                 ;REGISTER AND READ LOCATION
18262  110736  013767  177752  177206          MOV     @#HITMIS,RECDAT
18263  110744  032767  000004  177200  11$:    BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
18264  110752  001002                          BNE     12$                     ;THEN
18265  110754  000167  000254                  JMP     EXBAD2                  ;ERROR, EXIT
18266  110760  026767  177204  177204  12$:    CMP     LSTADD, CURADD          ;IF CURRENT ADDRESS IS LAST ADDRESS
18267  110766  001044                          BNE     16$                     ;THEN
18268  110770  016767  177166  177166          MOV     NEWDAT,CURDAT           ;NEW KIPAR5
18269  110776  022767  000001  177142          CMP     #1,     DCOUNT          ;IF LOOP COUNTER LESS THAN 1
18270  111004  003406                          BLE     13$                     ;THEN
18271  111006  006367  177146                  ASL     RITEDA                  ;UPDATE OF NEW DATA....
18272  111012  056767  177142  177142          BIS     RITEDA, NEWDAT          ;BY SETTING BITS
18273  111020  000421                          BR      15$                     ;ENDIF
18274  111022  022767  000001  177116  13$:    CMP     #1,     DCOUNT          ;CHANGE PATTERN?
18275  111030  001007                          BNE     14$                     ;IF SO, BRANCH
18276  111032  012767  022200  177120          MOV     #22200, RITEDA          ;START WITH THE SAME
18277  111040  012767  155400  177114          MOV     #155400,NEWDAT          ;DON'T DO ALL 1'S
18278  111046  000406                          BR      15$                     ;
18279  111050  006367  177104          14$:    ASL     RITEDA                  ;UPDATE OF NEW DATA....
18280  111054  046767  177100  177100          BIC     RITEDA, NEWDAT          ;BY CLEARING BITS
18281  111062  000400                          BR      15$                     ;ELSE
18282  111064  005267  177056          15$:    INC     DCOUNT                  ;INCREMENT THE LOOP COUNTER
18283  111070  016767  177072  177074          MOV     FSTADD, CURADD          ;FINISH FIRST CURRENT ADDRESS
18284  111076  000426                          BR      18$                     ;ELSE
18285  111100  005767  177050          16$:    TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18286  111104  001412                          BEQ     17$                     ;THEN
18287  111106  066767  177044  177056          ADD     ADDLSB, CURADD          ;ADD THE VALUE OF THE CURRENT LSB
18288  111114  020567  177052                  CMP     R5,     CURADD          ;IF CURRENT ADDRESS GREATER THAN HIGHEST
18289  111120  002015                          BGE     18$                     ;VIRTUAL ADDRESS THEN
18290  111122  162767  007776  177042          SUB     #7776,  CURADD          ;ROLL IT BACK
18291  111130  000411                          BR      18$                     ;ELSE
18292  111132  166767  177020  177032  17$:    SUB     ADDLSB, CURADD          ;IF ADDRESSING DOWNWARD THEN SUBTRACT LSB
18293  111140  020467  177026                  CMP     R4,     CURADD          ;IF CURRENT ADDRESS LESS THEN LOWEST
18294  111144  003403                          BLE     18$                     ;VIRTUAL ADDRESS THEN
18295  111146  062767  007776  177016          ADD     #7776,  CURADD          ;ROLL IT BACK
18296  111154  022767  000005  176764  18$:    CMP     #5,     DCOUNT          ;IF LOOP COUNTER LESS THAN 7
18297  111162  003402                          BLE     181$                    ;THEN LOOP BACK FOR NEXT BIT POSITION
18298  111164  000167  177416                  JMP     3$
18299  111170  005767  176760          181$:   TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18300  111174  001404                          BEQ     19$                     ;THEN
18301  111176  005067  176752                  CLR     FWDSEQ                  ;START ADDRESSING DOWNWARD
18302  111202  000167  177316                  JMP     TSLOOP                  ;ELSE
18303  111206  012767  000001  176740  19$:    MOV     #1,     FWDSEQ          ;START ADDRESSING UPWARD
18304  111214  006167  176736                  ROL     ADDLSB                  ;ROTATE LSB TO NEXT POSITION
18305  111220  022767  020000  176730          CMP     #20000, ADDLSB          ;ALL DONE?
18306  111226  001406                          BEQ     TSEND                   ;EXIT
```

```
 18307  111230  000167  177270                    JMP     TSLOOP                  ;ENDDO
 18308  111234  013701  172352          EXBAD2:   MOV     @#KIPAR5,R1             ;STORE PAR
 18309  111240  016700  176726                    MOV     CURADD, R0              ;AND ADDRESS
 18310  111244  012737  001200  172352  TSEND:    MOV     #1200,  @#KIPAR5        ;RESTORE PAR
 18311  111252  042737  000400  177520            BIC     #BIT08, @#BCSR          ;OUT OF STANDALONE MODE
 18312  111260  000207                            RTS     PC                      ;RETURN
 18313  111262                          ENDTAG:
 18314                                  .ENABL AMA
 18315
```

```
 18316                                  .SBTTL  TEST  - PCR READ/WRITE BITS
 18317                                  ;PCR AND BCSR READ/WRITE BITS
 18318                                  ;THE FIRST TEST WILL CHECK THAT PCR REGISTER IS BOTH WORD AND
 18319                                  ;BYTE ADDRESSABLE. BITS 14-09 AND 06-01 WILL BE WRITTEN AND READ
 18320                                  ;AS ZEROES AND ONES. THE REST OF THE BITS HAVE TO BE ALL 0'S.
 18321                                  ;ROUTINE TEST
 18322                                  ;.      SAVE PCR
 18323                                  ;.      LET PCR=0
 18324                                  ;.      DO FOR PATTERN=001111,110011,101010,010101
 18325                                  ;.      .       WRITE PCR<14-09>=PATTERN
 18326                                  ;.      .       WRITE PCR<06-01>=PATTERN
 18327                                  ;.      .       IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PREVIOUS
 18328                                  ;.      .                                              PATTERN
 18329                                  ;.      .       .       THEN ERROR
 18330                                  ;.      .       ENDIF
 18331                                  ;.      .       WRITE PCR<06-01>=PATTERN
 18332                                  ;.      .       IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PATTERN
 18333                                  ;.      .       .       THEN ERROR
 18334                                  ;.      .       ENDIF
 18335                                  ;.      ENDDO
 18336                                  ;.      WRITE PCR=0101010101010101
 18337                                  ;.      IF PCR NE 0101010001010100 THEN
 18338                                  ;.      .       ERROR
 18339                                  ;.      ENDIF
 18340                                  ;ENDROUTINE
 18341
 18342                                  ;;*****************************************************************
 18343  111262  000004                 TST30:  SCOPE
 18344  111264  005037  177522                 CLR     PCR                             ;INITIALIZE PCR TO 0
 18345  111270  012702  111427                 MOV     #SIXBIT+1,R2                    ;R2->TABLE OF PATTERNS FOR 6 R/W BITS IN
 18346  111274  012704  111426                 MOV     #SIXBIT,R4                      ;R3 POINTER TO PREVIOUS PATTERN
 18347  111300  012703  000004                 MOV     #4,R3                           ;DO 4 TIMES
 18348                                  ;
 18349                                  ; WRITE TO HIGH BYTE FIRST
 18350                                  ;
 18351  111304  111237  177523         1$:     MOVB    (R2),PCR+1                      ;WRITE TO HIGH BYTE
 18352  111310  121237  177523                 CMPB    (R2),PCR+1                      ;BYTE WRITTEN OK?
 18353  111314  001003                         BNE     2$                              ;IF NOT, BRANCH
 18354  111316  121437  177522                 CMPB    (R4),PCR                        ;LOW BYTE CHANGED?
 18355  111322  001405                         BEQ     3$                              ;IF NOT, BRANCH
 18356  111324  111237  001125         2$:     MOVB    (R2),$GDDAT+1                   ;EXPECTED PATTERN HIGH BYTE
 18357  111330  111437  001124                 MOVB    (R4),$GDDAT                     ;LOW BYTE
 18358  111334  104051                         ERROR   +51                             ;ERROR PCR READ/WRITE BITS
 18359  111336  105724                 3$:     TSTB    (R4)+                           ;INCREMENT POINTER FOR OLD
 18360                                  ;
 18361                                  ; WRITE TO LOW BYTE
 18362                                  ;
 18363  111340  111237  177522                 MOVB    (R2),PCR                        ;WRITE TO LOW BYTE
 18364  111344  121237  177522                 CMPB    (R2),PCR                        ;BYTE WRITTEN OK?
 18365  111350  001003                         BNE     4$                              ;IF NOT,BRANCH
 18366  111352  121237  177522                 CMPB    (R2),PCR                        ;HIGH BYTE CHANGED?
 18367  111356  001405                         BEQ     5$                              ;IF NOT, BRANCH
 18368  111360  111237  001124         4$:     MOVB    (R2),$GDDAT                     ;EXPECTED PATTERN HIGH BYTE
 18369  111364  111237  001124                 MOVB    (R2),$GDDAT                     ;LOW BYTE
 18370  111370  104051                         ERROR   +51                             ;ERROR PCR READ/WRITE BITS
 18371  111372  105722                 5$:     TSTB    (R2)+                           ;INCREMENT POINTER FOR NEW PATTERN
```

```
 18372  111374  077335                          SOB     R3,1$                           ;DO FOR ALL 4 PATTERNS
 18373                                  ;
 18374                                  ; NOW TRY WORD ADDRESSING
 18375                                  ;
 18376  111376  012737  052525  177522          MOV     #52525,PCR                      ;WRITE A PATERN
 18377  111404  022737  052124  177522          CMP     #52124,PCR                      ;ALL BUT BITS <8,7,0> OK?
 18378  111412  001404                          BEQ     6$                              ;IF SO, BRANCH
 18379  111414  112737  052124  001124          MOVB    #52124,$GDDAT                   ;EXPECTED PATTERN
 18380  111422  104051                          ERROR   +51                             ;ERROR PCR READ/WRITE BITS
 18381  111424                          6$:
 18382  111424  000403                          BR      TST31           ;;GO TO NEXT TEST
 18383
 18384  111426     000     036     146  SIXBIT: .BYTE 0,36,146,124,52                   ;001111,110011,101010,010101
 18385  111431     124     052
 18386                                                                                  ;BINARY DIVIDE FOR 6 BITS
 18387          111434                          .EVEN
 18388
```

```
18389                                   .SBTTL  TEST  - BCSR READ/WRITE BITS
18390                                   ;THE SECOND TEST WILL CHECK THAT BCSR<7-5;2-0> CAN BE WRITTEN AND
18391                                   ;READ AS ZEROES AND ONES.  BCSR<14,03> SHOULD BE 0'S.  BCSR<04>
18392                                   ;SHOULD BE CLEARED BY RESET INSTRUCTION.
18393                                   ;ROUTINE TEST
18394                                   ;.      FOR PATTERN=011,010,101 DO
18395                                   ;.      .       WRITE BCSR<7-5>=PATTERN
18396                                   ;.      .       IF BCSR<7-5> NE PATTERN THEN
18397                                   ;.      .       .       ERROR
18398                                   ;.      .       ENDIF
18399                                   ;.      .       WRITE BCSR<2-0>=PATTERN
18400                                   ;.      .       IF BCSR<2-0> NE PATTERN THEN
18401                                   ;.      .       .       ERROR
18402                                   ;.      .       ENDIF
18403                                   ;.      ENDDO
18404                                   ;.      IF BCSR<14,03> NE <0,0> THEN
18405                                   ;.      .       ERROR
18406                                   ;.      ENDIF
18407                                   ;.      LET BCSR<04>=1
18408                                   ;.      IF BCSR<04> NE #1 THEN
18409                                   ;.      .       ERROR
18410                                   ;.      ENDIF
18411                                   ;.      EXECUTE "RESET"
18412                                   ;.      IF BCSR<04> NE 0 THEN
18413                                   ;.      .       ERROR
18414                                   ;.      ENDIF
18415                                   ;.      LET BCSR<04>=0 (THIS BIT IS WRITE ENABLE FOR EAROM)
18416                                   ;ENDROUTINE
18417
18418                                   ;;**************************************************************************
18419  111434  000004                  TST31:  SCOPE
18420  111436  013737  177520  002710          MOV     BCSR,SAVBR                      ;SAVE BCSR
18421  111444  005037  177520                  CLR     BCSR                            ;CLEAR BCSR
18422                                  ;
18423                                  ; WRITE TO BITS <7-5> AND <2-0>
18424                                  ;
18425  111450  012703  111704                  MOV     #THRBIT,R3                      ;POINTER FOR PATTERN TABLE
18426  111454  005037  001124                  CLR     $GDDAT                          ;CLEAR A LOCATION
18427  111460  012702  000003                  MOV     #3,R2                           ;DO FOR ALL PATTERNS
18428  111464  111337  177520          1$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <7-5>
18429  111470  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN
18430  111474  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
18431  111500  001401                          BEQ     2$                              ;IF SO, BRANCH
18432  111502  104052                          ERROR   ^52                             ;ERROR IN BCSR READ/WRITE BITS
18433  111504  111337  177520          2$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <2-0>
18434  111510  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN FOR ERRORS
18435  111514  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
18436  111520  001401                          BEQ     3$                              ;IF SO, BRANCH
18437  111522  104052                          ERROR   ^52                             ;ERROR IN BCSR READ/WRITE BITS
18438  111524  077221                  3$:     SOB     R2,1$                           ;CONTINUE TILL ALL PATTERNS DONE
18439                                  ;
18440                                  ; CHECK UNUSED BITS <3>
18441                                  ;
18442  111526  012737  000010  177520          MOV     #10,BCSR                        ;WRITE TO BIT <3>
18443  111534  032737  000010  177520          BIT     #BIT03,BCSR                     ;ALL ZEROES?
18444  111542  001403                          BEQ     4$                              ;IF YES, BRANCH
```

```
COKDAAO KDJ11-B CLUSTER MACY11 30(1046)  23-JAN-84  18:56  PAGE 345
COKDAA.P11    23-JAN-84 18:55           TEST  - BCSR READ/WRITE BITS                                    SEQ 0345

 18445  111544  005037  001124                  CLR     $GDDAT                          ;EXPECTED PATTERN
 18446  111550  104052                          ERROR   ^52                             ;ERROR IN BCSR READ/WRITE BITS
 18447                                  ;
 18448                                  ; CHECK THAT BIT <4> CLEARS BY RESET
 18449                                  ;
 18450  111552  052737  000020  177520  4$:     BIS     #BIT04,BCSR                     ;SET BIT 4
 18451  111560  032737  000020  177520          BIT     #BIT04,BCSR                     ;WRITTEN OK?
 18452  111566  001005                          BNE     5$                              ;IF SO BRANCH
 18453  111570  012737  000020  001124          MOV     #BIT04,$GDDAT                   ;EXPECTED PATTERN
 18454  111576  104052                          ERROR   ^52                             ;ERROR IN BCSR READ/WRITE BITS
 18455  111600  000421                          BR      7$                              ;EXIT TEST
 18456  111602  042737  000020  177520  5$:     BIC     #BIT04,BCSR                     ;TRY TO CLEAR BIT 4
 18457  111610  032737  000020  177520          BIT     #BIT04,BCSR                     ;CLEARED OK?
 18458  111616  001403                          BEQ     6$                              ;IF SO BRANCH
 18459  111620  005037  001124                  CLR     $GDDAT                          ;EXPECTED PATTERN
 18460  111624  104052                          ERROR   ^52                             ;ERROR IN BCSR READ/WRITE BITS
 18461  111626  122737  000001  001220  6$:     CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
 18462  111634  001003                          BNE     7$                              ;NO, GO DO TEST
 18463  111636  005737  001206                  TST     $PASS                           ;FIRST PASS?
 18464  111642  001014                          BNE     8$                              ;IF APT AND NOT FIRST PASS,EXIT
 18465  111644  052737  000020  177520  7$:     BIS     #BIT04,BCSR                     ;SET BIT 4 AGAIN
 18466  111652  000005                          RESET                                   ;EXECUTE RESET
 18467  111654  032737  000020  177520          BIT     #BIT04,BCSR                     ;BIT 4 CLEARED?
 18468  111662  001404                          BEQ     8$                              ;IF YES, BRANCH
 18469  111664  104053                          ERROR   ^53                             ;RESET DOESN'T CLEAR BCSR<4>
 18470  111666  042737  000020  177520          BIC     #BIT04,BCSR                     ;CLEAR BIT 4
 18471  111674  013737  002710  177520  8$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
 18472  111702  000403                          BR      TST32           ;;GO TO NEXT TEST
 18473
 18474  111704     140     003     100  THRBIT: .BYTE   140,3,100,2,240,5               ;011,010,101 FOR BITS <7-5,2-0>
 18475  111707     002     240     005
 18476
```

```
18477                                   .SBTTL  TEST  - 16 BIT ROM CHECKSUM TEST
18478                                   ;ROM'S CHECKSUMS
18479                                   ;
18480                                   ;16 BIT ROM TEST
18481                                   ;THE FIRST TEST WILL CLEAR BCSR<07>, LOAD PCR<14-09> WITH
18482                                   ;ROM ADDRESS BITS <14-09>, AND CHECK CHECKSUMS OF 16-BIT ROM
18483                                   ;BY ACCESSING IT THRU BUS ADDRESSES 173000-173776.  THEN WITH
18484                                   ;BCSR<06;05> BOTH CLEAR, PCR<06-01> USED AS ADDRESS BITS 14-09,
18485                                   ;THE SAME THING WILL BE DONE BY ADDRESSING 16-BIT ROM THRU BUS ADDRESSES
18486                                   ;165000-165776. THE RESULTS SHOULD BE THE SAME AND SHOULD COMPARE
18487                                   ;WITH THAT STORED IN THE BOOT AND DIAGNOSTIC ROM.
18488                                   ;
18489                                   ;BCSR   <07>    DISABLE 17773000
18490                                   ;       <06>    DISABLE 17765000
18491                                   ;       <05>    ROM SOCKET 3 AT 17765000
18492                                   ;
18493                                   ;ROUTINE TEST
18494                                   ;.      LET BCSR<7,6,5>=0,0,0
18495                                   ;.      DO FOR R1 FROM #0 TO #31. BY #1 DO
18496                                   ;.      .       LET PCR<14-09>=R1
18497                                   ;.      .       DO FOR R2 FROM #0 TO #776 BY 2
18498                                   ;.      .       .       CALCULATE CHECKSUM THRU 173000
18499                                   ;.      .       ENDDO
18500                                   ;.      ENDDO
18501                                   ;.      IF CHECKSUM NE #0 THEN
18502                                   ;.      .       ERROR
18503                                   ;.      ENDIF
18504                                   ;.      DO FOR R1 FROM #0 TO #31. BY #1
18505                                   ;.      .       LET PCR<06-01>=R1
18506                                   ;.      .       DO FOR R2 FROM #0 TO #776 BY 2
18507                                   ;.      .       .       CALCULATE CHECKSUM THRU 165000
18508                                   ;.      .       ENDDO
18509                                   ;.      ENDDO
18510                                   ;.      IF CHECKSUM NE #0 THEN
18511                                   ;.      .       ERROR
18512                                   ;.      ENDIF
18513                                   ;ENDROUTINE
18514
18515                                   ;;*****************************************************************
18516  111712  000004                  TST32:  SCOPE
18517  111714  013737  177520  002710          MOV     BCSR,SAVBR                      ;SAVE BCSR
18518  111722  042737  000340  177520          BIC     #BIT07!BIT06!BIT05,BCSR         ;READ 16 BIT ROM
18519                                   ;
18520                                   ; CALCULATE LOW BYTE CHECKSUM'S THRU 173000 AND 165000
18521                                   ;
18522  111730  005001                          CLR     R1                              ;PAGE COUNT FOR ALL 8K
18523  111732  005037  177522                  CLR     PCR                             ;CLEAR PAGE CONTROL REGISTER
18524  111736  005037  002704          1$:     CLR     ACTCHS                          ;CLEAR CHECKSUM AT 173000
18525  111742  005037  001160                  CLR     $TMP0                           ;AT 165000
18526  111746  005002                          CLR     R2                              ;CLEAR COUNTER THRU A PAGE
18527  111750  016203  173000          2$:     MOV     173000(R2),R3                   ;GET LOW BYTE THRU 173000
18528  111754  016204  165000                  MOV     165000(R2),R4                   ;THRU 165000
18529  111760  060337  002704                  ADD     R3,ACTCHS                       ;CALCULATE CHECKSUM THRU 173000
18530  111764  060437  001160                  ADD     R4,$TMP0                        ;CALCULATE CHEKCSUM THRU 165000
18531  111770  005722                          TST     (R2)+                           ;GET NEXT WORD
18532  111772  022702  000776                  CMP     #776,R2                         ;WORD BEFORE LAST?
```

```
18533  111776  001004                          BNE     3$                      ;IF NOT, BRANCH
18534  112000  020337  177522                  CMP     R3,PCR                  ;PAGE NUMBER STORED OK?
18535  112004  001401                          BEQ     3$                      ;IF YES, BRANCH
18536  112006  104054                          ERROR   .54                     ;PAGE NUMBER STORED WRONG
18537  112010  022702  001000          3$:     CMP     #1000,R2                ;LAST WORD IN A PAGE?
18538  112014  003355                          BGT     2$                      ;IF NOT, BRANCH
18539  112016  005737  002704                  TST     ACTCHS                  ;CHECKSUM 0?
18540  112022  001401                          BEQ     4$                      ;IF YES, BRANCH
18541  112024  104054                          ERROR   .54                     ;IN CHECKSUM AT 173000
18542  112026  005737  001160          4$:     TST     $TMP0                   ;CHECKSUM 0 AT 165000?
18543  112032  001401                          BEQ     5$                      ;IF YES, BRANCH
18544  112034  104054                          ERROR   .54                     ;IN CHECKSUM AT 165000
18545  112036  062701  000002          5$:     ADD     #2,R1                   ;GET NEW PAGE
18546  112042  110137  177522                  MOVB    R1,PCR                  ;STORE IN PCR<6-1>
18547  112046  110137  177523                  MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
18548  112052  022701  000160                  CMP     #160,R1                 ;ALL  PAGES IN R2 FROM BIT1?
18549  112056  003327                          BGT     1$                      ;IF NOT BRANCH
18550                                          ;
18551                                          ; CHECK THE LAST 2K OF ASCII TEXT
18552                                          ;
18553  112060  005037  002704          LASTCH: CLR     ACTCHS                  ;CLEAR CHECKSUM AT 173000
18554  112064  005037  001160                  CLR     $TMP0                   ;AT 165000
18555  112070  005002                  1$:     CLR     R2                      ;CLEAR COUNTER THRU A PAGE
18556  112072  016203  173000          2$:     MOV     173000(R2),R3           ;GET LOW BYTE THRU 173000
18557  112076  016204  165000                  MOV     165000(R2),R4           ;THRU 165000
18558  112102  060337  002704                  ADD     R3,ACTCHS               ;CALCULATE CHECKSUM THRU 173000
18559  112106  060437  001160                  ADD     R4,$TMP0                ;CALCULATE CHEKCSUM THRU 165000
18560  112112  005722                          TST     (R2)+                   ;GET NEXT WORD
18561  112114  022702  001000          3$:     CMP     #1000,R2                ;LAST WORD IN A PAGE?
18562  112120  003364                          BGT     2$                      ;IF NOT, BRANCH
18563  112122  022701  000176                  CMP     #176,R1                 ;LAST PAGE?
18564  112126  001005                          BNE     4$                      ;IF NOT, BRANCH
18565  112130  023737  173774  177522          CMP     173774,PCR              ;PROPER PAGE NUMBER?
18566  112136  001401                          BEQ     4$                      ;IF YES, BRANCH
18567  112140  104054                          ERROR   .54                     ;ERROR IN ROM
18568  112142  062701  000002          4$:     ADD     #2,R1                   ;GET NEW PAGE
18569  112146  110137  177522                  MOVB    R1,PCR                  ;STORE IN PCR<6-1>
18570  112152  110137  177523                  MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
18571  112156  022701  000200                  CMP     #200,R1                 ;ALL  PAGES IN R2 FROM BIT1?
18572  112162  003342                          BGT     1$                      ;IF NOT BRANCH
18573  112164  005737  002704                  TST     ACTCHS                  ;CHECKSUM 0?
18574  112170  001401                          BEQ     5$                      ;IF YES, BRANCH
18575  112172  104054                          ERROR   .54                     ;IN CHECKSUM AT 173000
18576  112174  005737  001160          5$:     TST     $TMP0                   ;CHECKSUM 0 AT 165000?
18577  112200  001401                          BEQ     6$                      ;IF YES, BRANCH
18578  112202  104054                          ERROR   .54                     ;IN CHECKSUM AT 165000
18579  112204  005037  177522          6$:     CLR     PCR                     ;CLEAR PCR
18580  112210  01373.  177520  002710          MOV     BCSR,SAVBR              ;RESTORE BCSR
18581
18582
18583
```

```
                                        .SBTTL  TEST  - 8 BIT ROM CRC TEST
                                        ;THIS TEST WILL CLEAR BCSR<6>, SET BCSR<5>, LOAD PCR<5-1>
                                        ;WITH ADDRESS BITS 13-09, AND CHECK CRC PATTERNS OF 8-BIT ROM BY
                                        ;ACCESSING IT THRU ADDRESSES 165000-165776. THIS TEST VERIFIES THE
                                        ;RESPONSE ONLY OF THE BASE AREA.
                                        ;ROUTINE TEST
                                        ;.      LET BCSR<5>=1
                                        ;.      LET OLDCRC=#0
                                        ;.      LET PCR<05-01>=#0
                                        ;.      CALCULATE CHECKSUM FOR THE FIRST 320 LOCATIONS
                                        ;.      IF RESULTING CHECKSUM NOT ZERO THEN
                                        ;.              ERROR
                                        ;.      ENDIF
                                        ;ENDROUITNE

                                        ;;*****************************************************************
112216  000004                  TST33:  SCOPE
112220  013737  177520  002710          MOV     BCSR,SAVBR              ;SAVE BCSR
112226  042737  000100  177520          BIC     #BIT06,BCSR             ;ENABLE INTERNAL RESPONSE
112234  052737  000040  177520          BIS     #BIT05,BCSR             ;SELECT 8-BIT ROM
112242  005037  001160                  CLR     $TMP0                   ;CLEAR SUM
                                        ;
                                        ; CALCULATE LOW BYTE CHECKSUM'S THRU 165000
                                        ;
112246  005001                  1$:     CLR     R1                      ;PAGE COUNT FOR ALL 8K
112250  005037  177522                  CLR     PCR                     ;CLEAR PAGE CONTROL REGISTER
112254  005037  002704          2$:     CLR     ACTCHS                  ;CLEAR CHECKSUM AT 165000
112260  005002                          CLR     R2                      ;CLEAR COUNTER THRU A PAGE
112262  116204  165000          3$:     MOVB    165000(R2),R4           ;GET A BYTE THRU 165000
112266  060437  001160                  ADD     R4,$TMP0                ;CALCULATE CHEKCSUM THRU 165000
112272  005722                          TST     (R2)+                   ;GET NEXT WORD
112274  022702  000316                  CMP     #316,R2                 ;WORD BEFORE LAST?
112300  001004                          BNE     4$                      ;IF NOT, BRANCH
112302  122704  000252                  CMPB    #252,R4                 ;314 SHOULD HAVE 252
112306  001765                          BEQ     3$                      ;IF YES, BRANCH
112310  104055                          ERROR   #55                     ;PAGE NUMBER STORED WRONG
112312  022702  000322          4$:     CMP     #322,R2                 ;LAST WORD IN A PAGE?
112316  003361                          BGT     3$                      ;IF NOT, BRANCH
112320  113737  165010  001162          MOVB    165010,$TMP1            ;STORE SIZE,<3>=1 2K
112326  105737  001160                  TSTB    $TMP0                   ;CHECKSUM 0 AT 165000?
112332  001401                          BEQ     5$                      ;IF YES, BRANCH
112334  104055                          ERROR   #55                     ;IN CHECKSUM AT 165000
112336  005037  177522          5$:     CLR     PCR
112342  013737  002710  177520          MOV     SAVBR,BCSR              ;RESTORE BCSR
```

```
                                .SBTTL  TEST  - LKS BIT 7
                                ;THESE TESTS WILL CHECK FUNCTIONALITY OF LKS<7-6>
                                ;
                                ;ACCESS LKS, CHECK THAT READY LKS<07> CAN BE 0 AND 1.
                                ;
                                ;LKS     <07>    LINE CLOCK MONITOR
                                ;        <06>    LINE CLOCK INTERRUPT ENABLE
                                ;ROUTINE TEST
                                ;IF UFD AND LKS IS DISABLED THEN
                                ;.       EXIT TEST
                                ;ENDIF
                                ;.       READ LKS TO SEE IF IT TIMEOUTS
                                ;.(CHECK LKS<07>)
                                ;.       LET R1=#77777(WORST CASE COUNTER FOR SLOW CLOCK)
                                ;.       LET 100=#ADDRESS OF LINE_CLOCK_INTERRUPT
                                ;.       CLEAR INTERRUPT_FLAG
                                ;.       DO 3 TIMES
                                ;.       .       REPEAT
                                ;.       .       .       DECREMENT R1
                                ;.       .       UNTIL R1 NE #0 OR LOW BYTE OF LKS LT #0
                                ;.       .       IF LOW BYTE OF LKS GE #0 THEN (READY DIDN'T COME UP)
                                ;.       .       .       ERROR
                                ;.       .       ENDIF
                                ;.       ENDDO
                                ;.       IF INTERRUPT_FLAG NE #0 THEN (INTERUPT W/O LKS<6>=1)
                                ;.       .       ERROR
                                ;.       ENDIF
                                ;ENDROUTINE
                                ;
                                ;;*****************************************************************
112350  000004                  TST34:  SCOPE
112352  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
112360  001404                  1$      BEQ     1$                              ;IF NOT, GO DO TEST
112362  032737  010000  177520          BIT     #BIT12,BCSR                     ;LKS DISABLED?
112370  001053                          BNE     TST35           ;;IF DISABLED, EXIT TEST
                                ;
                                ; TRY TO ACCESS LKS WITHOUT TIMEOUT
                                ;
112372  013737  000004  001160  1$:     MOV     ERRVEC,$TMP0                    ;SAVE TIMEOUT VECTOR
112400  012737  112422  000004          MOV     #2$,ERRVEC                      ;POINT NEW VECTOR TO PROGRAM
112406  012737  000340  000006          MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
112414  005737  177546                  TST     LKS                             ;ACCESS LKS
112420  000403                          BR      3$                              ;IF NO TIMEOUT, CONTINUE
112422  104056                  2$:     ERROR   +56                             ;TIMEOUT READING LKS
112424  005726                          TST     (SP)+                           ;ADJUST STACK POINTER
112426  005726                          TST     (SP)+
112430  013737  001160  000004  3$:     MOV     $TMP0,ERRVEC                    ;RESTORE TIMEOUT VECTOR
                                ;
                                ; CHECK THAT READY BIT LKS<7> CAN BE 1
                                ;
112436  005037  002702                  CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
112442  012737  131650  000100          MOV     #LKSINT,100                     ;POINT VECTOR TO INERRUPT ROUTINE
112450  012737  000340  000102          MOV     #340,102                        ;AT PRIORITY 7
112456  012702  000003                  MOV     #3,R2                           ;DO 3 TIMES TO SYNCHRONISE
112462  012701  077777          4$:     MOV     #77777,R1                       ;COUNTER FOR SLOW CLOCKS
112466  105737  177546          5$:     TSTB    LKS                             ;READY LKS<7>=1?
```

```
 18686  112472  100401                        BMI     6$                              ;IF SO, DO NEXT LOOP
 18687  112474  077104                        SOB     R1,5$                           ;OTHERWISE, GO THRU COUNT
 18688  112476  105737  177546          6$:   TSTB    LKS                             ;WAS READY 1?
 18689  112502  100401                        BMI     7$                              ;IF YES, BRANCH
 18690  112504  104057                        ERROR   +57                             ;LKS<07> DOES NOT BECOME 1
 18691  112506  077213                  7$:   SOB     R2,4$                           ;DO ALL 3 TIMES
 18692  112510  005737  002702          8$:   TST     LKSFL                           ;ANY INTERRUPTS W/O LKS<6>=1?
 18693  112514  001401                        BEQ     TST35           ;;IF NONE, EXIT TEST
 18694  112516  104061                        ERROR   +61                             ;ILLEGAL CLOCK INTERRUPTS
 18695
```

```
18696                                   .SBTTL  TEST  - LKS INTERRUPT PRIORITY
18697                                   ;CHECK THAT LKS INTERRUPTS HAPPEN AT PRIORITY 5 CLEARING LKS<07>
18698                                   ;AND DON'T HAPPEN AT PRIORITY 6.
18699                                   ;ROUTINE TEST
18700                                   ;IF UFD AND LKS IS DISABLED THEN
18701                                   ;.      EXIT TEST
18702                                   ;ENDIF
18703                                   ;.      SET PRIORITY TO 5
18704                                   ;.      CLEAR INTERRUPT_FLAG
18705                                   ;.      LET LKS<06>=#1 (ENABLE INTERRUPTS)
18706                                   ;.      SET COUNTER TO WAIT FOR 3 INTERRUPTS
18707                                   ;.      REPEAT
18708                                   ;.      .       DECREMENT COUNTER
18709                                   ;.      UNTIL INTERRUPT_FLAG EQ #3 OR COUNTER EQ #0
18710                                   ;.      CLEAR LKS<06>
18711                                   ;.      IF LKS<07> EQ #1 THEN
18712                                   ;.      .       ERROR (WAS NOT CLEARED ON INTERRUPT)
18713                                   ;.      ENDIF
18714                                   ;.      IF COUNTER LT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR 800HZ
18715                                   ;.      .       ERROR (INTERRUPTS NEVER GO LOW)
18716                                   ;.      ENDIF
18717                                   ;.      IF INTERRUPT_FLAG LT #3 THEN
18718                                   ;.      .       ERROR (INTERRUPTS DON'T HAPPEN)
18719                                   ;.      ENDIF
18720                                   ;.      CLEAR INTERUPT_FLAG
18721                                   ;.      WAIT FOR LKS<7>=1
18722                                   ;.      LET LKS<7>=0
18723                                   ;.      IF LKS<7> NE #0 THEN
18724                                   ;.      .       ERROR (LKS<7> NOT CLEARED
18725                                   ;.      ENDIF
18726                                   ;.      SET PRIORITY TO 6
18727                                   ;.      SET COUNTER TO 1 SLOW CLOCK INTERRUPT
18728                                   ;.      SET LKS<06>
18729                                   ;.      REPEAT
18730                                   ;.      .       DECREMENT COUNTER
18731                                   ;.      UNTIL COUNTER EQ #0 OR INTERRUPT_FLAG NE #0
18732                                   ;.      IF INTERRUPT_FLAG NE #0 THEN
18733                                   ;.      .       ERROR (INTERRUPT WAS AT WRONG PRIORITY)
18734                                   ;.      ENDIF
18735                                   ;.      RESTORE ORIGINAL PRIORITY
18736                                   ;ENDROUTINE
18737                                   ;
18738                                   ;ROUTINE LINE_CLOCK_INTERUPT
18739                                   ;.      INCREMENT INTERRUPT_FLAG
18740                                   ;ENDROUTINE
18741
18742                                   ;;*****************************************************************
18743  112520  000004                  TST35:  SCOPE
18744  112522  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
18745  112530  001404                          BEQ     1$                              ;IF NOT, GO DO TEST
18746  112532  032737  010000  177520          BIT     #BIT12,BCSR                     ;LKS DISABLED?
18747  112540  001121                          BNE     TST36           ;;IF DISABLED, EXIT TEST
18748                                  ;
18749                                  ; WAIT FOR 3 INTERRUPTS AND CHECK LKS<7> TO BE 0 AFTER INTERRUPT
18750                                  ;
18751  112542  005037  002702          1$:     CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
```

```
18752  112546  012737  131650  000100          MOV     #LKSINT,100             ;POINT VECTOR TO ROUTINE
18753  112554  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
18754  112560  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
18755  112566  032737  000100  177546          BIT     #BIT06,LKS              ;BIT SET OK?
18756  112574  001001                          BNE     2$                      ;IF YES, BRANCH
18757  112576  104131                          ERROR   •131                    ;ERROR WRITING 1 TO LKS<6>
18758  112600  106427  000240          2$:     MTPS    #240                    ;SET PRIORITY TO 5
18759  112604  022737  000003  002702  3$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
18760  112612  001401                          BEQ     4$                      ;IF YES, BRANCH
18761  112614  077105                          SOB     R1,3$                   ;STAY IN A LOOP
18762                                  ;
18763                                  ; DISABLE INTERRUPTS AND CHECK THAT PROPER CONDITIONS ARE MET
18764                                  ;
18765  112616  106427  000340          4$:     MTPS    #340                    ;RAISE PRIORITY
18766  112622  042737  000100  177546          BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
18767  112630  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED AFTER INTERRUPTS?
18768  112636  001401                          BEQ     5$                      ;IF 0, BRANCH
18769  112640  104062                          ERROR   •62                     ;INTERRUPTS DON'T CLEAR LKS<7>
18770  112642  105737  177546          5$:     TSTB    LKS                     ;LKS<7>=1?
18771  112646  100375                          BPL     5$                      ;IF NOT, WAIT
18772  112650  005037  177546                  CLR     LKS                     ;CLEAR LKS<7>
18773  112654  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED?
18774  112662  001401                          BEQ     6$                      ;IF YES, BRANCH
18775  112664  104060                          ERROR   •60                     ;LKS<7> NOT CLEARED ON WRITE
18776  112666  032737  000100  177546  6$:     BIT     #BIT06,LKS              ;LKS<6>=0?
18777  112674  001401                          BEQ     7$                      ;IF YES, BRANCH
18778  112676  104131                          ERROR   •131                    ;ERROR WRITING 0 TO LKS<6>
18779  112700  022701  077737          7$:     CMP     #77737,R1               ;COUNTER AT LESS THAN 800HZ?
18780  112704  002001                          BGE     8$                      ;IF NOT, BRANCH
18781  112706  104063                          ERROR   •63                     ;READY LINE DOES NOT GO LOW
18782  112710  022737  000003  002702  8$:     CMP     #3,LKSFL                ;DID 3 INTERRUPTS HAPPEN?
18783  112716  001404                          BEQ     9$                      ;IF YES, BRANCH
18784  112720  012737  000003  001124          MOV     #3,$GDDAT               ;3 INTERRUPTS EXPECTED
18785  112726  104064                          ERROR   •64                     ;INTERRUPTS DON'T HAPPEN
18786                                  ;
18787                                  ; CHECK WHETHER INTERRUPTS HAPPEN AT PRIORITY 6
18788                                  ;
18789  112730  005037  002702          9$:     CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
18790  112734  106427  000300                  MTPS    #300                    ;RAISE PRIORITY TO 6
18791  112740  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
18792  112744  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
18793  112752  005737  002702          10$:    TST     LKSFL                   ;ANY INTERRUPTS?
18794  112756  001001                          BNE     11$                     ;IF YES, EXIT LOOP
18795  112760  077104                          SOB     R1,10$                  ;CONTINUE WITH COUNT
18796  112762  005737  002702          11$:    TST     LKSFL                   ;ANY INTERRUPTS?
18797  112766  001404                          BEQ     12$                     ;IF NO, BRANCH
18798  112770  012737  000005  001124          MOV     #5,$GDDAT               ;STORE PRIORITY FOR TYPE OUT
18799  112776  104065                          ERROR   •65                     ;INTERUPTS HAPPEN AT WRONG PRIORITY
18800  113000  106427  000340          12$:    MTPS    #340                    ;RESTORE PRIORITY
18801
```

```
                                        .SBTTL  TEST  - LINE CLOCK DISABLE
                                        ;LINE CLOCK DISABLE(*)
                                        ;THIS TEST WILL CHECK THAT BCSR<12> DISABLES RESPONSE
                                        ;OF LKS REGISTER.
                                        ;
                                        ;BCSR   <12>    LINE CLOCK STATUS REGISTER DISABLE
                                        ;ROUTINE TEST
                                        ;IF UFD AND LKS IS NOT DISABLED THEN
                                        ;.      EXIT TEST
                                        ;ENDIF
                                        ;.      WRITE BCSR<12>=1
                                        ;.      LET 4=ADDRESS OF LKS_TRAP
                                        ;.      LET TRAP_LKS=0
                                        ;.      READ LKS
                                        ;.      IF TRAP_LKS NE 1 THEN
                                        ;.              ERROR
                                        ;.      ENDIF
                                        ;ENDROUTINE
                                        ;
                                        ;ROUTINE LKS_TRAP
                                        ;.      LET TRAP_LKS=1
                                        ;.      LET BCSR<12>=0
                                        ;RTI

                                        ;;*****************************************************************
113004  000004                          TST36:  SCOPE
113006  032737  000040  000052                  BIT     @#BIT05,@#52                    ;UFD MODE?
113014  001404                                  BEQ     1$                              ;IF NOT, BRANCH
113016  032737  010000  177520                  BIT     @#BIT12,BCSR                    ;LKS DISABLED?
113024  001052                                  BNE     TST37           ;;IF DISABLED, EXIT TEST
                                        ;
                                        ; CHECK BCSR<12> TO BE 0 AND 1
                                        ;
113026  013737  177520  002710          1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
113034  042737  010000  177520                  BIC     @#BIT12,BCSR                            ;CLEAR BCSR
113042  032737  010000  177520                  BIT     @#BIT12,BCSR                    ;<12>=0?
113050  001403                                  BEQ     2$                              ;IF OK, BRANCH
113052  005037  001124                          CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
113056  104052                                  ERROR   ^52                             ;ERROR BCSR READ/WRITE BITS
113060  052737  010000  177520          2$:     BIS     @#BIT12,BCSR                    ;SET BIT 12
113066  032737  010000  177520                  BIT     @#BIT12,BCSR                    ;GOT SET OK?
113074  001004                                  BNE     3$                              ;IF OK, BRANCH
113076  012737  010000  001124                  MOV     @#BIT12,$GDDAT                  ;EXPECTED PATTERN
113104  104052                                  ERROR   ^52                             ;ERROR BCSR READ/WRITE BITS
                                        ;
                                        ; TRY TO ACCESS LKS TO GET A TIMEOUT WITH BCSR<12>=1
                                        ;
113106  013701  000004                  3$:     MOV     ERRVEC,R1                       ;SAVE TIMEOUT VECTOR
113112  012737  113134  000004                  MOV     @4$,ERRVEC                      ;POINT TO PROGRAM AREA
113120  012737  000340  000006                  MOV     @#340,ERRVEC+2                  ;AT PRIORITY 7
113126  005737  177546                          TST     LKS                             ;ACCESS LKS REGISTER
113132  104066                                  ERROR   ^66                             ;BCSR<12> DOES NOT DISABLE LKS
113134  005726                          4$:     TST     (SP)+                           ;RESTORE STACK POINTER
```

SEQ 0354

```
 18855  113136  005726                          TST     (SP)+
 18856  113140  010137  000004                  MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
 18857  113144  013737  002710  177520          MOV     SAVBR,BCSR              ;RESTORE BCSR
 18858
 18859
```

```
                                        .SBTTL  TEST  - UNCONDITIONAL CLOCK LINE INTERRUPTS
                                        ;UNCONDITIONAL CLOCK LINE INTERRUPTS(*)
                                        ;THIS TEST WILL CHECK THAT SETTING BCSR<13> TO 1 WILL
                                        ;REQUEST INTERRUPTS WHENEVER A CLOCK LINE IS ASSERTED.  THIS
                                        ;SHOULD HAPPEN WITHOUT ACCESSING LKS, THEREFORE, EVEN WITH BCSR<12>=1
                                        ;INTERRUPTS SHOULD HAPPEN.
                                        ;
                                        ;BCSR    <13>    FORCE LINE CLOCK INTERRUPT ENABLE
                                        ;

                                        ;ROUTINE TEST
                                        ;IF UFD AND FORCE LKS NOT DISABLED THEN
                                        ;.      EXIT TEST
                                        ;ENDIF
                                        ;.      LET 100=ADDRESS OF UNCONDITIONAL_INTERRUPT_ROUTINE
                                        ;.      DO FOR BCSR<12> FROM #0 TO #1(LKS DISABLED AND ENABLED)
                                        ;.      (IF UFD DO ONLY FOR SELECTED LINE CLOCK)
                                        ;.      .       CLEAR UNCONDITIONAL_INTERRUPT
                                        ;.      .       SET COUNTER TO WAIT  FOR 3 INTERRUPTS
                                        ;.      .       LET BCSR<13>=1
                                        ;.      .       REPEAT
                                        ;.      .       .       DECREMENT COUNTER
                                        ;.      .       UNTIL UNCONDITIONAL_INTERUPT EQ #3 OR COUNTER EQ #0
                                        ;.      .       IF COUNTER GT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR_800HZ
                                        ;.      .       .       ERROR (INTERRUPTS NEVER GO LOW)
                                        ;.      .       ENDIF
                                        ;.      .       IF UNCONDITIONAL_INTERRUPT LT #3 THEN
                                        ;.      .       .       ERROR (INTERRUPTS DON'T HAPPEN)
                                        ;.      .       ENDIF
                                        ;.      LET BCSR<13>=#0
                                        ;.      ENDDO
                                        ;ENDROUTINE
                                        ;
                                        ;ROUTINE UNCONDITIONAL_INTERRUPT_ROUTINE
                                        ;.      INCREMENT UNCONDITIONAL_INTERRUPT
                                        ;RETURN

                                        ;;*****************************************************************
113152  000004                  TST37:  SCOPE
113154  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
113162  001404                          BEQ     1$                              ;IF NOT, BRANCH
113164  032737  020000  177520          BIT     #BIT13,BCSR                     ;FORCE INTERRUPT SET?
113172  001525                          BEQ     TST40           ;;IF SET, EXIT TEST
                                        ;
                                        ; CHECK BSCR<13> TO BE 0 AND 1
                                        ;
113174  013737  177520  002710  1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
113202  042737  020000  177520          BIC     #BIT13,BCSR                     ;CLEAR BCSR
113210  032737  020000  177520          BIT     #BIT13,BCSR                     ;<13>=0?
113216  001403                          BEQ     2$                              ;IF OK, BRANCH
113220  005037  001124                  CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
113224  104052                          ERROR   .52                             ;ERROR BCSR READ/WRITE BITS
113226  052737  020000  177520  2$:     BIS     #BIT13,BCSR                     ;SET BIT 13
113234  032737  020000  177520          BIT     #BIT13,BCSR                     ;GOT SET OK?
113242  001004                          BNE     3$                              ;IF OK, BRANCH
113244  012737  020000  001124          MOV     #BIT13,$GDDAT                   ;EXPECTED PATTERN
```

```
18916  113252  104052                          ERROR   ^52                     ;ERROR BCSR READ/WRITE BITS
18917                                          ;
18918                                          ; SET UP  TO DO UNCONDITIONAL INTERRUPTS
18919                                          ;
18920  113254  012737  131650  000100  3$:     MOV     #LKSINT,@#100           ;SET UP INTERRUPT VECTOR
18921  113262  012737  000340  000102          MOV     #340,@#102              ;AT PRIORITY 7
18922  113270  052737  010000  177520          BIS     #BIT12,BCSR             ;FOR 1ST TIME DISABLE LKS
18923  113276  000403                          BR      5$                      ;GO DO IT
18924  113300  042737  010000  177520  4$:     BIC     #BIT12,BCSR             ;FOR THE 2ND TIME, ENABLE LKS
18925  113306  005037  002702          5$:     CLR     LKSFL                   ;CLEAR INTERRUPTS FLAG
18926  113312  012702  077777                  MOV     #77777,R2               ;COUNTER TO WAIT FOR INTERRUPTS
18927  113316  106427  000240                  MTPS    #240                    ;LOWER PRIORITY TO 5
18928  113322  022737  000003  002702  6$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
18929  113330  001401                          BEQ     7$                      ;EXIT LOOP, IF SO
18930  113332  077205                          SOB     R2,6$                   ;OTHERWISE, KEEP WAITING
18931  113334  106427  000340          7$:     MTPS    #340                    ;RAISE PRIORITY TO 7
18932  113340  022702  077700                  CMP     #77700,R2               ;INTERRUPTS HAPPEN TOO OFTEN?
18933  113344  002001                          BGE     8$                      ;IF NOT, BRANCH
18934  113346  104063                          ERROR   ^63                     ;READY LINE DOESN'T GO LOW
18935  113350  022737  000003  002702  8$:     CMP     #3,LKSFL                ;AT LEAST 3 INTERRUPTS HAPPENED?
18936  113356  002004                          BGE     9$                      ;IF SO, BRANCH
18937  113360  012737  000003  001124          MOV     #3,$GDDAT               ;EXPECTED DATA
18938  113366  104064                          ERROR   ^64                     ;INTERRUPTS DON'T HAPPEN
18939  113370  032737  010000  177520  9$:     BIT     #BIT12,BCSR             ;SECOND TIME THRU THE LOOP?
18940  113376  001340                          BNE     4$                      ;IF NOT, DO IT AGAIN
18941  113400  032737  000040  000052          BIT     #BIT05,@#52             ;UFD MODE?
18942  113406  001404                          BEQ     10$                     ;IF NOT, BRANCH
18943  113410  032737  010000  177520          BIT     #BIT12,BCSR             ;IF UFD AND LKS DISABLED?
18944  113416  001010                          BNE     12$                     ;DON'T CHECK LKS
18945  113420  032737  000100  177546  10$:    BIT     #BIT06,LKS              ;INTERRUPT ENABLE LINE HOLD 1?
18946  113426  001001                          BNE     11$                     ;IF SO, BRANCH
18947  113430  104067                          ERROR   ^67                     ;BCSR<13> DOESN'T SET LKS<6>
18948  113432  042737  000100  177546  11$:    BIC     #BIT06,LKS              ;DISABALE LKS INTERRUPTS
18949  113440  013737  002710  177520  12$:    MOV     SAVBR,BCSR              ;RESTORE BCSR
18950
18951
```

```
 18952                                  .SBTTL  TEST  - RESETTING LKS
 18953                                  ;RESETTING LKS(*)
 18954                                  ;THIS TEST WILL PROVE THAT RESET INSTRUCTION SETS LKS<07> AND
 18955                                  ;CLEARS LKS<06>.
 18956                                  ;ROUTINE TEST
 18957                                  ;IF UFD AND LKS IS DISABLED THEN
 18958                                  ;.      EXIT TEST
 18959                                  ;ENDIF
 18960                                  ;.      POINT LKS VECTOR 100 TO ERROR_LKS_ILLEGAL_INTERRUPT
 18961                                  ;.      SYNCHRONIZE LKS BY WAITING FOR 3 PULSES
 18962                                  ;.      LET LKS<06>=#1
 18963                                  ;.      CLEAR LKS (CLEARS LKS<07>
 18964                                  ;.      EXECUTE "RESET"
 18965                                  ;.      IF LKS<7> NE #1 OR LKS<6> NE #0 THEN
 18966                                  ;.      .       ERROR
 18967                                  ;.      ENDIF
 18968                                  ;.      IF ILLEGAL_LINE_CLOCK_INTERRUPT NE 0 THEN
 18969                                  ;.      .       ERROR
 18970                                  ;.      ENDIF
 18971                                  ;ENDROUTINE
 18972                                  ;
 18973                                  ;ROUTINE ERROR_LKS_ILLEGAL_INTERRUPT
 18974                                  ;.      FLAG ILLEGAL_LINE_CLOCK_INTERRUPT
 18975                                  ;RETURN
 18976                                  ;
 18977                                  ;;*****************************************************************
 18978  113446  000004                 TST40:  SCOPE
 18979  113450  122737  000001  001220         CMPB    @APTENV,$ENV                    ;RUNNING IN APT MODE?
 18980  113456  001013                         BNE     1$                              ;NO, GO DO TEST
 18981  113460  005737  001206                 TST     $PASS                           ;FIRST PASS?
 18982  113464  001057                         BNE     TST41           ;;IF APT AND NOT FIRST PASS, EXIT TEST
 18983  113466  032737  000040  000052         BIT     @BIT05,@#52                     ;UFD MODE?
 18984  113474  001404                         BEQ     1$                              ;IF NOT, BRANCH
 18985  113476  032737  010000  177520         BIT     @BIT12,BCSR                     ;LKS DISABLED?
 18986  113504  001447                         BEQ     TST41           ;;IF DISABLED, EXIT TEST
 18987                                  ;
 18988                                  ; SYNCHRONISE WITH LINE TIME CLOCK BY WAITING FOR 3 INTERRUPTS
 18989                                  ;
 18990  113506  013737  177520  002710 1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR
 18991  113514  042737  010000  177520         BIC     @BIT12,BCSR                     ;ENABLE LKS RESPONSE
 18992  113522  012737  131650  000100         MOV     @LKSINT,@#100                   ;SET UP INTERRUPT VECTOR
 18993  113530  012737  000340  000102         MOV     @340,@#102                      ;AT PROIRITY 7
 18994  113536  052737  000100  177546         BIS     @BIT06,LKS                      ;SET INTERRUPT ENABLE BIT
 18995  113544  005037  002702                 CLR     LKSFL                           ;CLEAR INTERRUPTS FLAG
 18996  113550  012702  077777                 MOV     @77777,R2                       ;COUNTER TO WAIT FOR INTERRUPTS
 18997  113554  106427  000240                 MTPS    @240                            ;LOWER PRIORITY TO 5
 18998  113560  022737  000003  002702 2$:     CMP     @3,LKSFL                        ;3 INTERRUPTS HAPPENED?
 18999  113566  001401                         BEQ     3$                              ;EXIT LOOP, IF SO
 19000  113570  077205                         SOB     R2,2$                           ;OTHERWISE, KEEP WAITING
 19001  113572  106427  000340         3$:     MTPS    @340                            ;RAISE PRIORITY TO 7
 19002  113576  000005                         RESET                                   ;EXECUTE RESET
 19003  113600  032737  000200  177546         BIT     @BIT07,LKS                      ;READY BIT SET?
 19004  113606  001001                         BNE     4$                              ;IF SO, BRANCH
 19005  113610  104070                         ERROR   .70                             ;RESET DOESN'T SET LKS<07.
 19006  113612  032737  000100  177546 4$:     BIT     @BIT06,LKS                      ;INTERRUPT ENABLE BIT CLEARED?
 19007  113620  001401                         BEQ     TST41           ;;IF SO, EXIT TEST
```

```
COKDAAO KDJ11-B CLUSTER MACY11 30(1046)  23-JAN-84  18:56  PAGE 358
COKDAA.P11    23-JAN-84 18:55           TEST  - RESETTING LKS                                                    SEQ 0358

 19008  113622  104071                          ERROR    +71                          ;RESET DOESN'T CLEAR LKS
 19009
 19010
```

```
19011                                   .SBTTL  TEST  - LINE CLOCK INTERRUPTS
19012                                   ;LINE CLOCK INTERRUPTS(*)
19013                                   ;BY SETTING TO 1 LKS<06>, THIS TEST WILL CHECK FOR INTERRUPTS
19014                                   ;FROM BEVENT LINE AND FROM KDJ11-B 50HZ, 60HZ, 800HZ ON BOARD SIGNALS
19015                                   ;(THE LATTER SIGNALS WILL BE ACCESSED BY SETTING BCSR<11-10>).
19016                                   ;
19017                                   ;BCSR   <11>    <10>            CLOCK SELECT BITS 1 AND 0
19018                                   ;
19019                                   ;        0       0              EXTERNAL BEVENT LINE
19020                                   ;        0       1              ON-BOARD 50 HZ
19021                                   ;        1       0              ON-BOARD 60 HZ
19022                                   ;        1       1              ON-BOARD 800 HZ
19023                                   ;
19024                                   ;ROUTINE TEST
19025                                   ;IF UFD THEN
19026                                   ;.      IF LKS DISABLED THEN
19027                                   ;.      .       EXIT TEST
19028                                   ;.      ENDIF
19029                                   ;.      SET FLAGS TO RUN ONLY WHAT SPECIFIED IN EAPROM
19030                                   ;ENDIF
19031                                   ;.      LET 100=ADDRESS OF LKS_INTERRUPT
19032                                   ;.      DO FOR BCSR<11;10> FROM #0 TO #3
19033                                   ;.      .       LET LKS<06>=#1
19034                                   ;.      .       WAIT FOR 10 INTERRUPTS FOR EACH CLOCK
19035                                   ;.      .       STORE ACTUAL NUMBER OF INTERRUPTS FOR EACH CLOCK
19036                                   ;.      .       LET INTERRUPT_FLAG=0
19037                                   ;.      ENDDO
19038                                   ;.      COMPARE NUMBER OF INTERRUPTS FOR EACH CLOCK
19039                                   ;ENDROUTINE
19040                                   ;
19041                                   ;ROUTINE LKS_INTERRUPT
19042                                   ;.      INCREMENT INTERRUPT_FLAG
19043                                   ;RETURN
19044
19045                                   ;;*****************************************************************
19046  113624  000004                   TST41:  SCOPE
19047  113626  032737  000040  000052           BIT     #BIT05,@#52                     ;UFD MODE?
19048  113634  001411                           BEQ     1$                              ;IF NOT, GO DO THE TEST
19049  113636  032737  010000  177520           BIT     #BIT12,BCSR                     ;LKS IS DISABLED?
19050  113644  001120                           BNE     TST42           ;;IF DISABLED, EXIT TESTS
19051  113646  005002                           CLR     R2                              ;CLEAR R2 TO SET FLAGS
19052  113650  053702  177520                   BIS     BCSR,R2                         ;SET R2 ACCORDING TO BCSR
19053  113654  042702  171777                   BIC     #171777,R2                      ;LEAVE ONLY BITS <11-10>
19054                                   ;
19055                                   ; SETUP FOR INTERRUPTS, IN UFD MODE ONLY FROM THE CLOCK SPECIFIED IN BCSR<11-10>
19056                                   ;
19057  113660  013737  177520  002710   1$:     MOV     BCSR,SAVBR                      ;STORE BCSR
19058  113666  012737  131650  000100           MOV     #LKSINT,100                     ;SET UP LKS VECTOR
19059  113674  012737  000340  000102           MOV     #340,102                        ;AT PRIORITY 7
19060  113702  012705  114076                   MOV     #TIMDEL,R5                      ;POINTER TO DEL
19061  113706  012704  000004                   MOV     #4,R4                           ;R4 IS THE COUNTER FOR ALL CLOCKS
19062  113712  005037  177520                   CLR     BCSR                            ;DO FOR BEVENT LINE INTERRUPTS
19063  113716  000403                           BR      3$                              ;GO DO THE LOOP
19064  113720  062737  002000  177520   2$:     ADD     #2000,BCSR                      ;SET UP FOR THE NEXT CLOCK LINE
19065  113726  032737  000040  000052   3$:     BIT     #BIT05,@#52                     ;UFD MODE?
19066  113734  001402                           BEQ     4$                              ;IF NOT, BRANCH
```

```
19067  113736  010237  177520            MOV     R2,BCSR                 ;IN UFD, DO ONLY FOR SPECIFIED
19068  113742  005037  002702      4$:   CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
19069  113746  052737  000100  177546    BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19070  113754  012703  000010            MOV     #10,R3                  ;START COUNTER FOR 10 INTERURRUPTS
19071  113760  012701  177777      5$:   MOV     #177777,R1              ;START COUNTER TO WAIT FOR INTERRUPT
19072  113764  106427  000240            MTPS    #240                    ;LOWER PRIORITY TO 5
19073  113770  023703  002702      6$:   CMP     LKSFL,R3                ;NEW INTERRUPT HAPPENED?
19074  113774  001401                    BEQ     7$                      ;IF SO, EXIT WAIT LOOP
19075  113776  077104                    SOB     R1,6$                   ;OTHERWISE, KEEP WAITING
19076  114000  010125            7$:   MOV     R1,(R5)+                ;STORE DELAY FOR EACH CLOCK
19077  114002  032737  000040  000052    BIT     #BIT05,@#52             ;UFD MODE?
19078  114010  001026                    BNE     10$                     ;IF UFD, DON'T DO FOR ANY OTHER
19079  114012  077436                    SOB     R4,2$                   ;ALL LINE CLOCKS DONE?
19080                                    ;
19081                                    ; CHECK THE DELAY VALUES FOR ALL CLOCKS
19082                                    ;
19083  114014  106427  000340            MTPS    #340                    ;RAISE PRIORITY
19084  114020  042737  000100  177546    BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
19085  114026  012705  114076            MOV     #TIMDEL,R5              ;POINTER TO DELAY TABLE
19086  114032  021565  000006            CMP     (R5),6(R5)              ;DELAY FOR BEVENT AND 800HZ?
19087  114036  103401                    BLO     8$                      ;BEVENT IS NOT 800HZ
19088  114040  104064                    ERROR   +64                     ;WRONG # OF INTERRUPTS
19089  114042  005725            8$:   TST     (R5)+                   ;INCREMENT POINTER
19090  114044  021565  000002            CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 60HZ?
19091  114050  103401                    BLO     9$                      ;IF FIRST BIGGER, BRANCH
19092  114052  104064                    ERROR   +64                     ;WRONG # OF INTERRUPTS
19093  114054  005725            9$:   TST     (R5)+                   ;INCREMENT POINTER
19094  114056  021565  000002            CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 800HZ
19095  114062  103401                    BLO     10$                     ;IF FIRST BIGGER, BRANCH
19096  114064  104064                    ERROR   +64                     ;WRONG # OF INTERRUPTS
19097  114066  013737  002710  177520  10$:   MOV     SAVBR,BCSR              ;RESTORE BCSR
19098  114074  000404                    BR      TST42           ;;EXIT TEST
19099
19100  114076  000004            TIMDEL: .BLKW 4
19101
```

```
                                       .SBTTL  TEST  - MAINTENANCE REGISTER TEST
                                       ;MAINTENANCE REGISTER TEST
                                       ;THIS TEST WILL ADDRESS MAINTENANCE REGISTER AND CHECK BITS
                                       ;7-4 TO BE 0010, 2-1 TO BE 10, AND READ BITS 10-08, 03, 00
                                       ;FOR FUTURE USE.  THOSE BITS REPRESENT THE FOLLOWING SIGNALS:
                                       ;MULTIPROCESSOR SLAVE, UNIBUS SYSTEM, FPA AVAILABLE, HALT/TRAP
                                       ;OPTION, AND AC POWER OKAY.
                                       ;ROUTINE TEST
                                       ;.      IF MAINT. REG. BITS <7-4> NE 0010 OR <2-1> NE 10 THEN
                                       ;.          ERROR
                                       ;.      ENDIF
                                       ;.      READ MAINT.REG. BITS <10-08,03,00>
                                       ;ENDROUTINE

                                       ;;**********************************************************************
114106  000004                 TST42:  SCOPE
114110  032737  174000  177750         BIT     #174000,MAIREG                  ;UNUSED BITS ALL ZEROS?
114116  001401                         BEQ     1$                              ;IF OK, BRANCH
114120  104132                         ERROR   .132                            ;MAINTENANCE REGISTER ERROR
114122  032737  000044  177750  1$:    BIT     #44,MAIREG                      ;<5,2> SET ?
114130  001001                         BNE     2$                              ;IF SO, BRANCH
114132  104132                         ERROR   .132                            ;MAINTENANCE REGISTER ERROR
114134  032737  000322  177750  2$:    BIT     #322,MAIREG                     ;<7,6,4,1> CLEAR?
114142  001401                         BEQ     TST43           ;;IF YES, BRANCH
114144  104132                         ERROR   .132
```

```
                                .SBTTL  TEST  - SERIAL LINE UNIT REGISTERS
                                ;SERIAL LINE UNIT TEST(*)
                                ;BCR<2-0> WILL BE READ TO FIND OUT BAUD RATE. SLU WILL BE PROG-
                                ;RAMMED TO CHECK THE INTERRUPT LEVELS BY SETTING BIT<06> IN
                                ;RCSR AND XMIT. LOOP BACK CAPABILITIES WILL BE TESTED BY SETTING
                                ;TO 1 XCSR<02>.   THE LINE CLOCK INTERRUPT SUBROUTINE WILL BE
                                ;USED TO RETURN TO THE EXECUTION OF THE DIAGNOSTICS, IF THE
                                ;PROGRAM HANGS IN THE LOOP BACK MODE.
                                ;ROUTINE TEST
                                ;IF UFD AND CONSOLE NOT PRESENT
                                ;.      GO TO TEST_22
                                ;ENDIF
                                ;.      IF BCR<07> EQ #0 THEN
                                ;.      .       READ BCR<2-0> TO GET BAUD RATE
                                ;.      ENDIF
                                ;.      LET 4=ADDRESS_OF_TIMEOUT _ROUTINE
                                ;.      DO FOR RCSR,XCSR,RBUF,XBUF
                                ;.      .       READ XRCSR,XCSR,RBUF,XBUF
                                ;.      .       IF TIMEOUT_FLAG NE #0 THEN
                                ;.      .       .       ERROR
                                ;.      .       ENDIF
                                ;.      ENDDO
                                ;ENDROUTINE
                                ;
                                ;ROUTINE TIMEOUT
                                ;.      LET TIMEOUT_FLAG=#1
                                ;ENDROUTINE

                                ;;*****************************************************************
114146  000004                  TST43:  SCOPE
114150  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
114156  001406                        BEQ     1$                              ;IF NOT, GO DO THE TEST
114160  032737  000200  177524        BIT     #BIT07,BCR                      ;IF UFD AND CONSOLE NOT PRESENT
114166  001402                        BEQ     1$                              ;NOT TRUE, DO THE TEST
114170  000137  116044                JMP     SLEND                           ;IF TRUE, SKIP ALL SLU TESTS
                                ;
                                ; TRY TO ACCESS SLU REGISTERS
                                ;
114174  013701  000004          1$:     MOV     ERRVEC,R1                       ;SAVE TIMEOUT VECTOR
114200  012737  114224  000004        MOV     #3$,ERRVEC                      ;POINT NEW ONE TO PROGRAM AREA
114206  012737  000340  000006        MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
114214  012702  177560                MOV     #RCSR,R2                        ;START ACCESSING WITH RCSR
114220  005712                  2$:     TST     (R2)                            ;ACCESS SLU REGISTER
114222  000403                        BR      4$                              ;IF NO TIMEOUT, CONTINUE
114224  010237  001126          3$:     MOV     R2,@#BDDAT                      ;STORE ADDRESS THAT TIMED OUT
114230  104072                        ERROR   +72                             ;TIMEOUT ACCESSING SLU REGISTER
114232  022722  177566          4$:     CMP     #XBUF,(R2)+                     ;LAST REGISTER ACCESSED?
114236  103770                        BLO     2$                              ;IF NOT, BRANCH
114240  010137  000004                MOV     R1,ERRVEC                       ;RESTORE TIMEOUT VECTOR
```

```
 19179                                  .SBTTL  TEST  - XCSR BIT 7
 19180                                  ;CHECK THAT XCSR<07> CAN BE 0 AND 1.
 19181                                  ;
 19182                                  ;XCSR   <07>            TRANSMITTER READY
 19183                                  ;
 19184                                  ;ROUTINE TEST
 19185                                  ;.      WAIT FOR XCSR<07>=#1 NO MORE THAN 200MSEC
 19186                                  ;.      IF XCSR<07> NE #1 THEN
 19187                                  ;.          .       ERROR
 19188                                  ;.      ENDIF
 19189                                  ;.      LET XBUF=#NULL
 19190                                  ;.      WAIT FOR XCSR<07>=#1
 19191                                  ;.      LET XBUF=#NULL
 19192                                  ;.      IF XCSR<07> NE 0 THEN
 19193                                  ;.          .       ERROR (READY DIDN'T GO LOW)
 19194                                  ;.      ENDIF
 19195                                  ;ENDROUTINE
 19196
 19197                                  ;;*****************************************************************
 19198  114244  000004                 TST44:  SCOPE
 19199  114246  012701  001000                 MOV     #1000,R1                        ;COUNTER FOR ABOUT 200MICROSEC.
 19200  114252  105737  177564         1$:     TSTB    XCSR                    ;XCSR<7> READY 1?
 19201  114256  100401                         BMI     2$                      ;IF SO, EXIT WAIT LOOP
 19202  114260  077104                         SOB     R1,1$                   ;IF NOT 1, CONTINUE WAITING
 19203  114262  105737  177564         2$:     TSTB    XCSR                    ;XCSR<7>=1?
 19204  114266  100401                         BMI     3$                      ;IF YES, BRANCH
 19205  114270  104073                         ERROR   +73                     ;XCSR<7> DOES NOT BECOME 1
 19206  114272  012737  000000  177566 3$:     MOV     #NULL,XBUF              ;TRY TO TRANSMIT NULL CHARACTER
 19207  114300  105737  177564                 TSTB    XCSR                    ;XCSR<7>=0
 19208  114304  100001                         BPL     TST45           ;;IF YES, EXIT TEST
 19209  114306  104073                         ERROR   +73                     ;XMIT READY DIDN'T GO LOW
 19210
 19211
```

```
19212                                   .SBTTL  TEST  - RCSR BIT 7 AND XCSR BIT 2
19213                                   ;CHECK THAT RCSR<07> CAN BE 0 AND 1 AND THAT XCSR<02> WORKS PROPERLY.
19214                                   ;
19215                                   ;RCSR   <07>    RECEIVER DONE
19216                                   ;XCSR   <02>    MAINTENANCE
19217                                   ;
19218                                   ;ROUTINE TEST
19219                                   ;.(CHECK RCSR<07> AND XCSR<07>)
19220                                   ;.      WAIT FOR XCSR<07>=#1
19221                                   ;.      LET XCSR<02>=#1 (LOOP BACK MODE)
19222                                   ;.      LET XBUF=#125
19223                                   ;.      WAIT FOR RCSR<07>=#1 NO MORE THAN 200MSEC
19224                                   ;.      IF RCSR<07> NE #1 THEN
19225                                   ;.      .       ERROR (RCSR<07  DOES NOT BECOME 1 OR XCSR<02>DOES NOT
19226                                   ;.      .               WORK)
19227                                   ;.      ENDIF
19228                                   ;.      IF RBUF NE #125 THEN
19229                                   ;.      .       ERROR
19230                                   ;.      ENDIF
19231                                   ;.      IF RCSR<07> NE #0 THEN
19232                                   ;.      .       ERROR (RCSR<07>DOES NOT GO LOW)
19233                                   ;.      ENDIF
19234                                   ;.      LET XCSR<02>=#0
19235                                   ;ENDROUITNE
19236
19237                                   ;;*****************************************************************
19238  114310  000004                  TST45:  SCOPE
19239  114312  012701  000013                  MOV     #13,R1                  ;COUNTER FOR ABOUT 200MICROSEC.
19240  114316  105737  177564          1$:     TSTB    XCSR                    ;XCSR<7> READY 1?
19241  114322  100375                          BPL     1$                      ;IF NOT 1, CONTINUE WAITING
19242  114324  052737  000004  177564  2$:     BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
19243  114332  032737  000004  177564          BIT     #BIT02,XCSR             ;GOT SET OK?
19244  114340  001004                          BNE     3$                      ;IF YES, BRANCH
19245  114342  005037  177564                  CLR     XCSR                    ;RESET TO PRINT ERROR
19246  114346  104114                          ERROR   .114                    ;XCSR<2> DOES NOT BECOME 1
19247  114350  000456                          BR      TST46           ;;EXIT TEST
19248                                  ;
19249                                  ; STALL FOR A WHILE IN CASE XCSR<2> CAUSES RCSR<7> TO BE 1
19250                                  ;
19251  114352  012701  002000          3$:     MOV     #2000,R1                ;STALL IN CASE XCSR<2> SETS READY
19252  114356  105737  177560          4$:     TSTB    RCSR                    ;IF RECEIVER READY SET?
19253  114362  100402                          BMI     5$                      ;IF SET, BRANCH
19254  114364  077104                          SOB     R1,4$                   ;OTHERWISE, STAY FOR A WHILE
19255  114366  000402                          BR      6$                      ;IF NOT READY, BRANCH
19256  114370  005737  177562          5$:     TST     RBUF                    ;READ RBUF
19257                                  ;
19258                                  ; TRANSMIT XON AND CHECK RCSR<7>
19259                                  ;
19260  114374  012737  000021  177566  6$:     MOV     #21,XBUF                ;TRANSMIT A CHARACTER
19261  114402  012701  002000                  MOV     #2000,R1                ;COUNTER TO WAIT
19262  114406  105737  177560          7$:     TSTB    RCSR                    ;RCSR<7> READY 1?
19263  114412  100401                          BMI     8$                      ;IF YES, EXIT WAIT LOOP
19264  114414  077104                          SOB     R1,7$                   ;OTHERWISE, CONTINUE WAITING
19265  114416  105737  177560          8$:     TSTB    RCSR                    ;RCSR<7>=1?
19266  114422  100403                          BMI     9$                      ;IF YES, BRANCH
19267  114424  005037  177564                  CLR     XCSR                    ;RESET XCSR<2>
```

```
 19268  114430  104074                                ERROR   ;74                ;RECEIVER READY DIDN'T COME UP
 19269  114432  013737  177562  001126  9$:           MOV     RBUF,$BDDAT        ;STORE RECEIVED DATA
 19270  114440  022737  000021  001126                CMP     #21,$BDDAT         ;DATA RECEIVED OK?
 19271  114446  001406                                BEQ     10$                ;IF YES, BRANCH
 19272  114450  012737  000021  001124                MOV     #21,$GDDAT
 19273  114456  005037  177564                        CLR     XCSR               ;RESET TO ENABLE SLU
 19274  114462  104075                                ERROR   ;75                ;WRONG CHARACTER RECEIVED
 19275  114464  105737  177560          10$:          TSTB    RCSR               ;RCSR<7>=0?
 19276  114470  100003                                BPL     11$                ;IF ZERO, BRANCH
 19277  114472  005037  177564                        CLR     XCSR               ;RESET TO ENABLE SLU
 19278  114476  104076                                ERROR   ;76                ;RCSR<07><>0 AFTER READING RBUF
 19279  114500  042737  000004  177564  11$:          BIC     #BIT02,XCSR        ;DISABLE LOOP BACK MODE
 19280
 19281
```

```
                                        .SBTTL  TEST  - RESET AND XCSR<2!0>
                                        ;CHECK THAT RESET CLEARS XCSR<0!2>.
                                        ;ROUTINE TEST
                                        ;.(CHECK RCSR<07> AND XCSR<07> AND RESET)
                                        ;.      LET XCSR<02,00>=#1 (LOOP BACK MODE)
                                        ;.      EXECUTE "RESET"
                                        ;.      IF XCSR<02!00> NE #0 THEN
                                        ;.              ERROR
                                        ;.      ENDIF
                                        ;.      LET XCSR<02>=#0
                                        ;ENDROUTINE

                                        ;;*****************************************************************
114506  000004                          TST46:  SCOPE
114510  122737  000001  001220                  CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
114516  001003                                  BNE     1$                              ;NO, GO DO TEST
114520  005737  001206                          TST     $PASS                           ;FIRST PASS?
114524  001011                                  BNE     TST47           ;;IF APT AND NOT FIRST PASS, EXIT TEST
114526  052737  000005  177564  1$:             BIS     #BIT02!BIT00,XCSR               ;LOOP BACK MODE
                                        ;
                                        ; EXECUTE RESET AND VALIDATE THAT XCSR<7,2> BECOMES <1,0>
                                        ;
114534  000005                                  RESET                                   ;EXECUTE RESET
114536  032737  000005  177564                  BIT     #BIT02!BIT00,XCSR               ;XCSR<2,0> CLEAR?
114544  001401                                  BEQ     TST47           ;;IF YES, BRANCH
114546  104102                                  ERROR   +102                            ;XCSR<2,0> NOT CLEARED ON RESET
```

```
                                .SBTTL  TEST  - RESET AND INTERRUPT ENABLE BITS
                                ;CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY 4 AND THAT RESET
                                ;CLEARS XCSR<06> AND RCSR<06>.
                                ;
                                ;RCSR   <06>    RECEIVER INTERRUPT ENABLE
                                ;XCSR   <06>    TRANSMITTER INTERRUPT ENABLE
                                ;
                                ;ROUTINE TEST
                                ;.      LET 60=@ADDRESS_OF_ILLEGAL_INTERRUPT_XRCSR
                                ;.      LET 64=@ADDRESS_OF_ILLEGAL_INTERRUPT_XRCSR
                                ;.      SET PRIORITY TO 4
                                ;.      LET XCSR<02>=#1 (LOOPBACK MODE)
                                ;.      LET XCSR<06>=#1 (ENABLE TRANSMIT INTERRUPTS)
                                ;.      LET RCSR<06>=#1 (ENABLE RECEIVE INTERRUPTS)
                                ;.      WAIT FOR XCSR<07>=#1 (READY TO TRANSMIT)
                                ;.      LET XBUF=#NULL (SEND A CHARACTER)
                                ;.      WAIT FOR ILLEGAL INTERRUPTS (ABOUT 200MSEC)
                                ;.      EXECUTE "RESET"
                                ;.      IF XCSR<06> NE #0 OR RCSR<06> NE #0 OR XRCSR NE #0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      RESTORE PRIORITY TO NORMAL
                                ;ENDROUTINE
                                ;
                                ;ROUTINE ILLEGAL_INTERRUPT_XRCSR
                                ;.      INCREMENT XRCSR
                                ;ENDROUTINE

                                ;;*****************************************************************
114550  000004                  TST47:  SCOPE
114552  122737  000001  001220          CMPB    @APTENV,$ENV                    ;RUNNING IN APT MODE?
114560  001003                          BNE     1$                              ;NO, GO DO TEST
114562  005737  001206                  TST     $PASS                           ;FIRST PASS?
114566  001033                          BNE     5$                              ;SKIP RESET PART OF THE TEST
                                ;
                                ; CHECK THAT INTERRUPTS ENABLE BITS FOR RECEIVER AND TRASMITTER OF SLU
                                ; ARE CLEARED BY RESET
                                ;
114570  052737  000100  177564  1$:     BIS     @#BIT06,XCSR                    ;SET INTERRUPT ENABLE BIT IN XCSR
114576  032737  000100  177564          BIT     @#BIT06,XCSR                    ;GOT SET OK?
114604  001001                          BNE     2$                              ;IF YES, BRANCH
114606  104110                          ERROR   .110                            ;IN BIT 6 OF XCSR
114610  052737  000100  177560  2$:     BIS     @#BIT06,RCSR                    ;SET INTERRUPT ENABLE BIT IN RCSR
114616  032737  000100  177560          BIT     @#BIT06,RCSR                    ;GOT SET OK?
114624  001001                          BNE     3$                              ;IF YES, BRANCH
114626  104110                          ERROR   .110                            ;IN BIT 6 OF RSCR
114630  000005                  3$:     RESET                                   ;INLINE BUS RESET
114632  032737  000100  177564          BIT     @#BIT06,XCSR                    ;XMIT INTERRUPT ENABLE BIT CLEARED?
114640  001401                          BEQ     4$                              ;IF CLEARED, BRANCH
114642  104102                          ERROR   .102                            ;INTERRUPT ENABLE NOT CLEARED ON RESET
114644  032737  000100  177560  4$:     BIT     @#BIT06,RCSR                    ;RECEIVE INTERRUPT ENBLE CLEARED?
114652  001401                          BEQ     5$                              ;IF CLEARED, BRANCH
114654  104102                          ERROR   .102                            ;INTERRUPT ENABLE NOT CLEARED ON RESET
                                ;
                                ; CHECK THAT TRANSMIT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
                                ;
```

```
19366  114656  012737  114724  000064  5$:     MOV     #9$,@#64                ;POINT XMIT VECTOR TO PROGRAM AREA
19367  114664  012737  000340  000066          MOV     #340,@#66               ;AT PRIORITY 7
19368  114672  052737  000100  177564          BIS     #BIT06,XCSR             ;SET INTERRUPT ENABLE BIT IN XCSR
19369  114700  012702  000340                  MOV     #340,R2                 ;SET PRIORITY TO 7
19370  114704  000402                          BR      7$                      ;GO WAIT IN CASE OF INTERRUPTS
19371  114706  162702  000040          6$:     SUB     #40,R2                  ;LOWER PRIORITY LEVEL
19372  114712  106402                  7$:     MTPS    R2                      ;SET PRIORITY
19373  114714  012703  000026                  MOV     #26,R3                  ;TIME DELAY
19374  114720  077301                  8$:     SOB     R3,8$                   ;WAIT FOR INTERRUPTS
19375  114722  000403                          BR      10$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19376  114724  104101                  9$:     ERROR   +101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
19377  114726  005726                          TST     (SP)+                   ;CLEAN UP THE STACK
19378  114730  005726                          TST     (SP)+
19379  114732  022702  000200          10$:    CMP     #200,R2                 ;AT PRIORITY 4?
19380  114736  001363                          BNE     6$                      ;IF NOT LAST ONE, CONTINUE
19381  114740  106427  000340                  MTPS    #340                    ;RESTORE PRIORITY 7
19382  114744  042737  000100  177564          BIC     #BIT06,XCSR             ;CLEAR INTERRUPT ENABLE BIT
19383                                  ;
19384                                  ; CHECK THAT RECEIVE INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19385                                  ;
19386  114752  012737  115042  000060          MOV     #15$,@#60               ;POINT RECEIVE VECTOR TO PROGRAM AREA
19387  114760  012737  000340  000062          MOV     #340,@#62               ;AT PRIORITY 7
19388  114766  105737  177564          11$:    TSTB    XCSR                    ;TRANSMITTER READY
19389  114772  100375                          BPL     11$                     ;IF NOT, WAIT
19390  114774  012737  000000  177566          MOV     #NULL,XBUF              ;TRY TO TRANSMIT NULL
19391  115002  052737  000100  177560          BIS     #BIT06,RCSR             ;SET INTERRUPT ENABLE BIT IN RCSR
19392  115010  012702  000340                  MOV     #340,R2                 ;SET PRIORITY TO 7
19393  115014  000402                          BR      13$                     ;GO WAIT IN CASE OF INTERRUPTS
19394  115016  162702  000040          12$:    SUB     #40,R2                  ;LOWER PRIORITY LEVEL
19395  115022  052737  000004  177564  13$:    BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
19396  115030  106402                          MTPS    R2                      ;SET PRIORITY
19397  115032  012703  000100                  MOV     #100,R3                 ;TIME DELAY
19398  115036  077301                  14$:    SOB     R3,14$                  ;WAIT FOR INTERRUPTS
19399  115040  000406                          BR      16$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19400  115042  042737  000004  177564  15$:    BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
19401  115050  104101                          ERROR   +101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
19402  115052  005726                          TST     (SP)+                   ;CLEAN UP THE STACK
19403  115054  005726                          TST     (SP)+
19404  115056  022702  000200          16$:    CMP     #200,R2                 ;AT PRIORITY 4?
19405  115062  001355                          BNE     12$                     ;IF NOT LAST ONE, CONTINUE
19406                                  ;
19407                                  ; CLEAN UP BEFORE NEXT TEST
19408                                  ;
19409  115064  106427  000340                  MTPS    #340                    ;RESTORE PRIORITY 7
19410  115070  042737  000100  177560          BIC     #BIT06,RCSR             ;CLEAR INTERRUPT ENABLE BIT
19411  115076  012702  000300                  MOV     #300,R2                 ;STALL DELAY
19412  115102  105737  177560          17$:    TSTB    RCSR                    ;RECEIVE READY?
19413  115106  100401                          BMI     18$                     ;STOP WAITING, IF SO
19414  115110  077204                          SOB     R2,17$                  ;OTHERWISE, STAY IN THE LOOP
19415  115112  005737  177562          18$:    TST     RBUF                    ;READ CHARACTER TRANSMITTED
19416  115116  042737  000004  177564          BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
19417
```

```
 19418                                  .SBTTL  TEST  - INTERRUPT PRIORITY FOR SLU
 19419                                  ;CHECK THAT INTERRUPTS HAPPEN AT PRIORITY 3 AND THAT THEY CLEAR
 19420                                  ;RCSR<06> AND XCSR<06>.
 19421                                  ;
 19422                                  ;ROUTINE TEST
 19423                                  ;.      LET 60=@ADDRESS_OF_LEGAL_RINTERRUPT
 19424                                  ;.      LET 64=@ADDRESS_OF_LEGAL_XINTERRUPT
 19425                                  ;.      LET XCSR<02>=#1
 19426                                  ;.      SET PRIORITY TO #3
 19427                                  ;.      WAIT FOR XINTERRUPT=#3
 19428                                  ;.      IF XCSR<07> EQ #1 THEN
 19429                                  ;.      .       ERROR
 19430                                  ;.      ENDIF
 19431                                  ;.      WAIT FOR RINTERRUPT=#3
 19432                                  ;.      IF RCSR<07> EQ #0 THEN
 19433                                  ;.      .       ERROR
 19434                                  ;.      ENDIF
 19435                                  ;.      LET XCSR<02>=#0
 19436                                  ;.      SET PRIORITY TO NORMAL
 19437                                  ;ENDROUTINE
 19438                                  ;
 19439                                  ;ROUTINE LEGAL_XINTERRUPT
 19440                                  ;.      LET XBUF=@CHARACTER
 19441                                  ;.      INCREMENT XINTERRUPT
 19442                                  ;ENDROUTINE
 19443                                  ;
 19444                                  ;ROUTINE LEGAL_RINTERRUPT
 19445                                  ;.      READ RCSR
 19446                                  ;.      INCREMENT RINTERRUPT
 19447                                  ;ENDROUTINE
 19448
 19449                                  ;;*****************************************************************
 19450  115124  000004                  TST50:  SCOPE
 19451                                  ;
 19452                                  ; GET READY FOR INTERRUPTS
 19453                                  ;
 19454  115126  012737  115322  000060          MOV     #8$,@#60                ;STORE RECEIVER VECTOR
 19455  115134  012737  115246  000064          MOV     #6$,@#64                ;STORE TRANSMITTER VECTOR
 19456  115142  012737  000340  000062          MOV     #340,@#62               ;AT PRIORITY 7
 19457  115150  012737  000340  000066          MOV     #340,@#66               ;FOR RECEIVER AND TRANSMITTER
 19458  115156  052737  000004  177564          BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
 19459  115164  012701  000100                  MOV     #100,R1                 ;DELAY FOR UNEXPECTED CHARACTERS
 19460  115170  105737  177560          1$:     TSTB    RCSR                    ;RECEIVER READY?
 19461  115174  100401                          BMI     2$                      ;IF YES, BRANCH
 19462  115176  077104                          SOB     R1,1$                   ;OTHERWISE, WAIT JUST IN CASE
 19463  115200  005737  177562          2$:     TST     RBUF                    ;READ RECEIVER
 19464                                  ;
 19465                                  ; SET PRIORITIES AND XMIT INTERRUPTS
 19466                                  ;
 19467  115204  012702  000140                  MOV     #140,R2                 ;START WITH PRIORITY 3
 19468  115210  000402                          BR      4$                      ;TRY TO DO IT
 19469  115212  162702  000040          3$:     SUB     #40,R2                  ;LOWER PRIORITY
 19470  115216  106402                  4$:     MTPS    R2                      ;TRY TO DO AT LOWER PRIORITY
 19471  115220  052737  000100  177564          BIS     #BIT06,XCSR             ;LOOP BACK & INTERRUPT ENABLE
 19472  115226  012703  000100                  MOV     #100,R3                 ;WAIT DELAY FOR INTERRUPTS
 19473  115232  077301                  5$:     SOB     R3,5$                   ;WAIT FOR XMIT INTERRUPTS
```

```
19474  115234  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK BIT
19475  115242  104107                          ERROR   ^107                           ;NO XMIT INTERRUPTS
19476  115244  000443                          BR      TST51           ;;IF ERROR, EXIT TEST
19477                                  ;
19478                                  ; TRANSMITTER INTERRUPT HERE
19479                                  ;
19480  115246  005726                  6$:     TST     (SP)+                           ;CLEAN UP STACK
19481  115250  005726                          TST     (SP)+
19482  115252  042737  000100  177564          BIC     #BIT06,XCSR                     ;CLEAR INTERRUPT ENABLE
19483  115260  012737  000000  177566          MOV     #NULL,XBUF                      ;TRANSMIT NULL
19484  115266  052737  000100  177560          BIS     #BIT06,RCSR                     ;SET RECEIVE INTERRUPT
19485  115274  106402                          MTPS    R2                              ;SET NEXT PRIORITY
19486  115276  012703  020000                  MOV     #20000,R3                       ;WAIT DELAY FOR INTERRUPTS
19487  115302  077301                  7$:     SOB     R3,7$                           ;WAIT FOR RECEIVE INTERUPTS
19488  115304  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK MODE BIT
19489  115312  104107                          ERROR   ^107                           ;NO RECEIVE INTERRUPTS
19490  115314  000406                          BR      9$                              ;DON'T TOUCH STACK
19491  115316  106427  000340                  MTPS    #340                            ;RAISE PRIORITY
19492                                  ;
19493                                  ; RECEIVER INTERRUPT HERE
19494                                  ;
19495  115322  005726                  8$:     TST     (SP)+                           ;CLEAN UP STACK
19496  115324  005726                          TST     (SP)+
19497  115326  005737  177562                  TST     RBUF                            ;READ RECEIVER BUFFER
19498  115332  005702                  9$:     TST     R2                              ;PRIORITY 0
19499  115334  001326                          BNE     3$                              ;IF NOT YET, CONTINUE
19500  115336  106427  000340                  MTPS    #340                            ;RAISE PRIORITY TO 7
19501  115342  042737  000100  177560          BIC     #BIT06,RCSR                     ;CLEAR RECEIVE INTER. ENABLE
19502  115350  005037  177564                  CLR     XCSR                            ;CLEAR XCSR
19503
```

```
19504                                   .SBTTL  TEST  - BREAK CONDITION
19505                                   ;CHECK THAT SENDING BREAK CAUSES FRAMING ERROR.
19506                                   ;
19507                                   ;RCSR   <15>    ERROR
19508                                   ;       <13>    FRAMING ERROR
19509                                   ;       <11>    RECEIVED BREAK
19510                                   ;
19511                                   ;XCSR   <00>    TRANSMIT BREAK
19512                                   ;
19513                                   ;ROUTINE TEST
19514                                   ;.      LET XCSR<02>=#1
19515                                   ;.      LET XCSR<00>=#1
19516                                   ;.      WAIT FOR RCSR<07>=#1
19517                                   ;.      IF RBUF<15!13!11> NE #1 THEN
19518                                   ;.              ERROR (ERROR, FRAMING ERROR, RECEIVE BREAK NE 1)
19519                                   ;.      ENDIF
19520                                   ;.      LET XCSR<00>=#0
19521                                   ;.      IF XCSR<00> NE #0 THEN
19522                                   ;.              ERROR (XCSR<00> DOES NOT GO LOW)
19523                                   ;.      ENDIF
19524                                   ;.      WAIT FOR XCSR<07>=#1
19525                                   ;.      LET XBUF=#NULL (SEND NULL CHARACTER TO SEE ERROR CLEARED)
19526                                   ;.      WAIT FOR RCSR<07>=#1
19527                                   ;.      IF RBUF<15!13!11> NE #0 THEN
19528                                   ;.              ERROR
19529                                   ;.      ENDIF
19530                                   ;.      LET XCSR<00>=#1
19531                                   ;.      EXECUTE "RESET"
19532                                   ;.      IF XCSR<00> NE #0 THEN
19533                                   ;.              ERROR
19534                                   ;.      ENDIF
19535                                   ;.      LET XCSR<02>=#0
19536                                   ;ENDROUTINE
19537
19538                                   ;;**********************************************************************
19539   115354  000004                  TST51:  SCOPE
19540                                   ;
19541                                   ; DECIDE WHETHER TO RUN THIS TEST
19542                                   ;
19543   115356  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
19544   115364  001127                          BNE     TST52           ;;IN UFD MODE, EXIT TEST
19545   115366  005737  001206                  TST     $PASS                           ;FIRST PASS?
19546   115372  001124                          BNE     TST52           ;;IF APT AND NOT FIRST PASS, EXIT TEST
19547                                   ;
19548                                   ; SEND BREAK AND CHECK ERROR BITS IN RBUF
19549                                   ;
19550   115374  052737  000004  177564  1$:     BIS     #BIT02,XCSR                     ;TRANSMIT IN LOOP BACK
19551   115402  013737  177520  002710          MOV     BCSR,SAVBR                      ;SAVE BCSR
19552   115410  042737  001000  177520          BIC     #BIT09,BCSR                     ;DISABLE HALT ON BREAK
19553   115416  052737  000001  177564          BIS     #BIT00,XCSR                     ;SET SEND BREAK BIT
19554   115424  032737  000001  177564          BIT     #BIT00,XCSR                     ;GOT SET OK?
19555   115432  001001                          BNE     2$                              ;IF YES, BRANCH
19556   115434  104110                          ERROR   .110                            ;WRITING 1 TO XCSR<0>
19557   115436  012701  000100          2$:     MOV     #100,R1                         ;STALL DELAY
19558   115442  105737  177560          4$:     TSTB    RCSR                            ;RECEIVER READY?
19559   115446  100401                          BMI     5$                              ;IF YES, BRANCH
```

```
19560  115450  077104                          SOB     R1,4$                   ;WAIT JUST IN CASE OF A CHARACTER
19561  115452  005737  177562          5$:     TST     RBUF                    ;READ A CHARACTER
19562  115456  052737  000001  177564          BIS     #BIT00,XCSR             ;TRANSMIT BREAK
19563  115464  012701  001000                  MOV     #1000,R1                ;ANOTHER DELAY TO GET BREAK
19564  115470  077101                  6$:     SOB     R1,6$                   ;WAIT A WHILE
19565  115472  105737  177560          7$:     TSTB    RCSR                    ;RECEIVER READY?
19566  115476  100375                          BPL     7$                      ;IF NOT, WAIT
19567  115500  013737  177562  001126          MOV     RBUF,$BDDAT             ;STORE WHATEVER RECEIVED
19568  115506  022737  124000  001126          CMP     #BIT15!BIT13!BIT11,$BDDAT ;ALL ERROR BITS SET?
19569  115514  001405                          BEQ     8$                      ;IF YES, BRANCH
19570  115516  042737  000004  177564          BIC     #BIT02,XCSR             ;RESET TO ENABLE SLU
19571  115524  104105                          ERROR   ^105                    ;BREAK DOES NOT CAUSE ERRORS
19572  115526  000446                          BR      TST52           ;;EXIT
19573  115530  042737  000001  177564  8$:     BIC     #BIT00,XCSR             ;CLEAR TRANSMIT BREAK
19574  115536  032737  000001  177564          BIT     #BIT00,XCSR             ;GOT CLEARED OK?
19575  115544  001405                          BEQ     9$                      ;IF YES, BRANCH
19576  115546  042737  000004  177564          BIC     #BIT02,XCSR             ;RESET TO ENABLE SLU
19577  115554  104110                          ERROR   ^110                    ;ERROR WRITING 0 TO XCSR<0>
19578  115556  000432                          BR      TST52           ;;EXIT
19579
19580                                  ;
19581                                  ; CHECK THAT BREAK CONDITION IS CLEARED
19582                                  ;
19582  115560  013737  002710  177520  9$:     MOV     SAVBR,BCSR              ;RESTORE BCSR
19583  115566  105737  177564          10$:    TSTB    XCSR                    ;XMIT READY?
19584  115572  100375                          BPL     10$                     ;IF NOT, WAIT
19585  115574  012737  000177  177566          MOV     #177,XBUF               ;TRY TO TRANSMIT DELETE
19586  115602  105737  177560          11$:    TSTB    RCSR                    ;RECEIVER READY
19587  115606  100375                          BPL     11$                     ;IF NOT, WAIT
19588  115610  013737  177562  001126          MOV     RBUF,$BDDAT             ;STORE RECEIVE BUFFER
19589  115616  032737  124000  001126          BIT     #BIT15!BIT13!BIT11,$BDDAT ;ERRORS CLEARED?
19590  115624  001404                          BEQ     12$                     ;IF YES, BRANCH
19591  115626  042737  000004  177564          BIC     #BIT02,XCSR             ;RESET TO ENABLE SLU
19592  115634  104106                          ERROR   ^106                    ;BREAK NOT CLEARED ON NEXT CHARACTER
19593  115636  042737  000004  177564  12$:    BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
19594
19595
```

```
 19596                                  .SBTTL  TEST  - OVERRUN CONDITION
 19597                                  ;CHECK OVERRUN CONDITION
 19598                                  ;
 19599                                  ;RCSR   <14>    OVERRUN ERROR
 19600                                  ;
 19601                                  ;ROUTINE TEST
 19602                                  ;.      LET XCSR<02>=#1 (LOOPBACK MODE)
 19603                                  ;.      WAIT FOR XCSR<07>=#1
 19604                                  ;.      LET XBUF=#252
 19605                                  ;.      WAIT FOR XCSR<07>=#1
 19606                                  ;.      LET XBUF=#125 (SEND THE 2ND W/O READING THE 1ST CHARACTER)
 19607                                  ;.      WAIT FOR RCSR<07>=#1
 19608                                  ;.      STALL FOR LOWEST BAUD RATE TO GET 2ND CHARACTER
 19609                                  ;.      IF LOW BYTE OF RBUF NE #125 THEN
 19610                                  ;.              ERROR (1ST CHARACTER WASN'T OVERRUN)
 19611                                  ;.      ENDIF
 19612                                  ;.      IF RBUF<15!14> NE #1 THEN
 19613                                  ;.              ERROR (NO OVERRUN BIT SET)
 19614                                  ;.      ENDIF
 19615                                  ;.      WAIT FOR XCSR<07>=#1
 19616                                  ;.      LET XBUF=#NULL
 19617                                  ;.      WAIT FOR RCSR<07>=#1
 19618                                  ;.      IF RBUF<15!14> NE #0 THEN
 19619                                  ;.              ERROR (WASN'T CLEARED ON THE NEXT CHARACTER RECEIVED)
 19620                                  ;.      ENDIF
 19621                                  ;.      LET XCSR<02>=#0
 19622                                  ;ENDROUTINE
 19623
 19624                                  ;;**************************************************************************
 19625  115644  000004                 TST52:  SCOPE
 19626  115646  052737  000004  177564         BIS     #BIT02,XCSR                     ;SET LOOP BACK MODE
 19627  115654  105737  177564         1$:     TSTB    XCSR                            ;READY TO TRANSMIT?
 19628  115660  100375                         BPL     1$                              ;IF NOT, WAIT
 19629  115662  012737  000021  177566         MOV     #21,XBUF                        ;TRANSMIT A CHARACTER
 19630  115670  105737  177560         2$:     TSTB    RCSR                            ;RECEIVE READY?
 19631  115674  100375                         BPL     2$                              ;IF NOT, WAIT
 19632  115676  105737  177564         3$:     TSTB    XCSR                            ;READY TO TRANSMIT?
 19633  115702  100375                         BPL     3$                              ;IF NOT, WAIT
 19634  115704  012737  000177  177566         MOV     #177,XBUF                       ;TRANSMIT THE 2ND CHARACTER
 19635  115712  012703  010000                 MOV     #10000,R3                       ;STALL FOR THE 2ND CHARACTER
 19636  115716  077301                 4$:     SOB     R3,4$                           ;WAIT A WHILE
 19637  115720  013737  177562  001126         MOV     RBUF,$BDDAT                     ;STORE RECEIVED DATA
 19638  115726  012737  140177  001124         MOV     #140177,$GDDAT                  ;EXPETED PATTERN
 19639  115734  122737  000177  001126         CMPB    #177,$BDDAT                     ;2ND CHARACTER RECEIVED?
 19640  115742  001404                         BEQ     5$                              ;IF YES, BRANCH
 19641  115744  042737  000004  177564         BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
 19642  115752  104111                         ERROR   .111                            ;2ND CHARACTER DIDN'T OVERRUN 1ST
 19643  115754  122737  000300  001127  5$:     CMPB    #BIT7!BIT6,$BDDAT+1             ;OVERRUN ERROR BITS SET?
 19644  115762  001404                         BEQ     6$                              ;IF YES, BRANCH
 19645  115764  005037  177564                 CLR     XCSR                            ;RESET TO ENABLE SLU
 19646  115770  104112                         ERROR   .112                            ;OVERRUN DOES NOT SET ERRORS BITS
 19647  115772  000424                         BR      TST53           ;;EXIT
 19648                                  ;
 19649                                  ; SEND NEXT CHARACTER TO CLEAR OVERRUN CONDITIONS
 19650                                  ;
 19651  115774  105737  177564         6$:     TSTB    XCSR                            ;TRANSMITTER READY?
```

```
 19652  116000  100375                          BPL     6$                      ;IF NOT, BRANCH AND WAIT
 19653  116002  012737  000000  177566          MOV     #NULL,XBUF              ;TRANSMIT NULL CHARACTER
 19654  116010  105737  177560          7$:     TSTB    RCSR                    ;RECEIVER READY?
 19655  116014  100375                          BPL     7$                      ;IF NOT, BRANCH AND WAIT
 19656  116016  032737  140000  177562          BIT     #BIT15!BIT14,RBUF       ;ANY ERRORS SET?
 19657  116024  001404                          BEQ     8$                      ;IF NOT, BRANCH
 19658  116026  042737  000004  177564          BIC     #BIT02,XCSR             ;RESET TO ENABLE SLU
 19659  116034  104113                          ERROR   •113                    ;OVERRUN NOT CLEARED ON NEXT CHAR.
 19660  116036  042737  000004  177564  8$:     BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE BIT
 19661
 19662  116044                          SLEND:                                  ;LAST SLU TEST
```

```
 19663                                  .SBTTL  TEST  - LED'S ON
 19664                                  ;LED'S ON
 19665                                  ;THIS TEST WILL INITIALIZE BDR TO CONTAIN A ROTATING PATTERN
 19666                                  ;DISPLAYED IN LED'S.
 19667                                  ;
 19668                                  ;ROUTINE TEST
 19669                                  ;.      WHILE A KEY NOT RECEIVED FROM KEYBOARD DO
 19670                                  ;.      .       STALL ALLOWING TIME TO SEE PATTERN
 19671                                  ;.      .       ROTATE LEFT TO LIGHT UP NEXT LED'S
 19672                                  ;.      ENDDO
 19673                                  ;ENDROUTINE
 19674
 19675                                  ;;*****************************************************************
 19676  116044  000004                  TST53:  SCOPE
 19677  116046  005005                          CLR     R5                      ;FLAG IN NO INTERRRUPT MODE
 19678  116050  032737  000040  000052          BIT     #BIT05,@#52             ;IF RUNNING IN UFD MODE
 19679  116056  001027                          BNE     1$                      ;SKIP PRINTOUTS
 19680  116060  005737  001206                  TST     $PASS                   ;1ST PASS?
 19681  116064  001024                          BNE     1$                      ;IF NOT, SKIP PRINTOUTS
 19682  116066  122737  000001  001220          CMPB    @APTENV,$ENV            ;APT MODE?
 19683  116074  001420                          BEQ     1$                      ;YES, SKIP PRINTOUT'S
 19684  116076  005105                          COM     R5                      ;CLEAR FLAG IN INTERRUPT MODE
 19685  116100  104401  001175                  TYPE    .$CRLF
 19686  116104  104401  116226                  TYPE    .LEDS                   ;IDENTIFY THE TEST
 19687  116110  012737  116206  000060          MOV     #5$,@#60                ;RECEIVE SLU VECTOR
 19688  116116  012737  000340  000062          MOV     #340,@#62               ;AT PRIORITY 7
 19689  116124  052737  000100  177560          BIS     #BIT06,RCSR             ;ENABLE INTERRUPTS
 19690  116132  106427  000140                  MTPS    #140                    ;LOWER PRIORITY
 19691  116136  012704  000006          1$:     MOV     #6,R4                   ;FOR EACH LOOP
 19692  116142  012701  000076                  MOV     #76,R1                  ;START WITH 1
 19693  116146  110137  177524          2$:     MOVB    R1,BDR                  ;TURN OFF FIRST LED
 19694  116152  012703  000004                  MOV     #4,R3                   ;STALL DELAY
 19695  116156  012702  177777          3$:     MOV     #177777,R2              ;STALL DELAY
 19696  116162  077201                  4$:     SOB     R2,4$                   ;WAIT A WHILE
 19697  116164  077304                          SOB     R3,3$                   ;WAIT A WHILE
 19698  116166  000261                          SEC                             ;SET CARRY
 19699  116170  006101                          ROL     R1                      ;GET ANOTHER LED
 19700  116172  077413                          SOB     R4,2$                   ;DO A FEW TIMES
 19701  116174  005705                          TST     R5                      ;RUNNING IN INTERACTIVE MODE?
 19702  116176  001407                          BEQ     6$                      ;IF NOT, EXIT
 19703  116200  104401  001170                  TYPE    .$BELL                  ;
 19704  116204  000754                          BR      1$                      ;REPEAT PATTERN
 19705  116206  005737  177562          5$:     TST     RBUF                    ;READ BUFFER
 19706  116212  062706  000004                  ADD     #4,SP                   ;ADJUST STACK
 19707  116216  112737  000377  177524  6$:     MOVB    #377,BDR                ;NO MORE
 19708  116224  000452                          BR      TST54           ;;EXIT TEST
 19709
 19710  116226  006412  044124  051511  LEDS:   .ASCII  <12><15>/THIS IS A TEST FOR ON-BOARD LED'S/<12><15>
 19711  116234  044440  020123  020101
 19712  116242  042524  052123  043040
 19713  116250  051117  047440  026516
 19714  116256  047502  051101  020104
 19715  116264  042514  023504  005123
 19716  116272     015
 19717  116273     124  050131  020105          .ASCIZ  /TYPE ANY CHARACTER ON A KEYBOARD TO CONTINUE/<12><15>
 19718  116300  047101  020131  044103
```

```
 19719  116306  051101  041501  042524
 19720  116314  020122  047117  040440
 19721  116322  045440  054505  047502
 19722  116330  051101  020104  047524
 19723  116336  041440  047117  044524
 19724  116344  052516  005105  000015
 19725                          .EVEN
 19726
```

```
                                .SBTTL  TEST  - MEMORY MAPPING
                                ;MEMORY MAPPING
                                ;THIS TEST WILL AUTOSIZE MEMORY IN 2K BYTES. EVERY PAGE WILL BE
                                ;CHECKED FOR WHAT TYPE IT IS: Q-BUS, UNIBUS OR PMI BUS MEMORY BY
                                ;SEEING HOW IT CAN BE CACHED.
                                ;ROUTINE TEST
                                ;.  LET 4=ADDRESS OF NON-EXISTENT_MEMORY
                                ;.  IF KMCR<05-00> NE #1 THEN (NOT ALL UNIBUS MEMORY)
                                ;.      TURN ON MMU AND REMAP THE PROGRAM AREA
                                ;.      REPEAT
                                ;.      .  DO FOR KDPAR0 FROM #0 TO #177600
                                ;.      .       DO FOR R1 FROM #0 TO #20000 BY #4000
                                ;.      .       .       READ (R1)
                                ;.      .       .       IF ABORT_NON-EXISTENT_MEMORY NE 1
                                ;.      .       .       .       MEMORY EXISTS
                                ;.      .       .       .       READ (R1)+
                                ;.      .       .       .       READ (R1)
                                ;.      .       .       .       IF HIT/MISS EQ 2_HITS THEN
                                ;.      .       .       .       .       PMI MEMORY
                                ;.      .       .       .       ELSE
                                ;.      .       .       .       .       IF HIT/MISS EQ HIT THEN
                                ;.      .       .       .       .       .       Q-BUS MEMORY
                                ;.      .       .       .       .       ELSE
                                ;.      .       .       .       .       .       ERROR
                                ;.      .       .       .       .       ENDIF
                                ;.      .       .       .       ENDIF
                                ;.      .       .       ENDIF
                                ;.      .       .       LET ABORT_NON-EXISTENT_MEMORY=#0
                                ;.      .       ENDDO
                                ;.      .  ENDDO
                                ;.      UNTILL ABORT_NON-EXISTANT_MEMORY EQ #1
                                ;.  ENDIF
                                ;.
                                ;ENDROUTINE
                                ;
                                ;ROUTINE NON-EXISTENT_MEMORY
                                ;.      LET ABORT_NON-EXISTENT_MEMORY=#1
                                ;.      RETURN
                                ;ENDROUTINE
                                ;
                                ;;*****************************************************************
116352  000004                  TST54:  SCOPE
116354  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
116362  001153                        BNE     TST55           ;;IF SO, EXIT TEST
116364  005737  001206                TST     $PASS                           ;FIRST PASS?
116370  001150                        BNE     TST55           ;;IF APT AND NOT FIRST PASS, EXIT TEST
116372  122737  000001  001220        CMPB    @APTENV,$ENV                    ;APT MODE?
116400  001544                        BEQ     TST55           ;;YES, SKIP PRINTOUT'S
                                ;
                                ; SETUP ALL REGISTERS FOR MAPPING
                                ;
116402  012737  000400  177746        MOV     #400,CCR                        ;FLUSH THE CACHE
116410  013704  000004                MOV     ERRVEC,R4                       ;STORE TIMEOUT VECTOR
116414  012737  116550  000004        MOV     #71$,@#ERRVEC                   ;POINT TO PROGRAM AREA
116422  012737  000340  000006        MOV     #340,ERRVEC+2                   ;AT PRIORITY 4
116430  004737  131362                JSR     PC,INITMM                       ;REMAP PROGRAM AREA
```

```
 19783  116434  005037  172354                  CLR     KIPAR6                  ;USED FOR MAPPING MEMORY
 19784  116440  005002                          CLR     R2                      ;CLEAR PAGE COUNT FOR PMI
 19785  116442  005003                          CLR     R3                      ;CLEAR PAGE COUNT FOR QBUS
 19786  116444  005237  177572                  INC     MMR0                    ;ENABLE MEMORY MANGEMENT
 19787  116450  052737  000020  172516          BIS     #BIT04,MMR3             ;ENABLE 22 BITS
 19788  116456  000403                          BR      2$                      ;GO ACCESS
 19789                                  ;
 19790                                  ; TRY TO MAP ALL PAGES
 19791                                  ;
 19792  116460  062737  000200  172354  1$:     ADD     #200,KIPAR6             ;INCREMENT BY 4K WORDS
 19793  116466  012701  140000          2$:     MOV     #140000,R1              ;ACCESS THRU KIPAR6
 19794  116472  000402                          BR      4$                      ;START DOING IT
 19795  116474  062701  003776          3$:     ADD     #3776,R1                ;INCREMENT BY 1K WORDS
 19796  116500  005721                  4$:     TST     (R1)+                   ;ACCESS 1ST LOCATION
 19797  116502  005711                          TST     (R1)                    ;ACCESS 2ND LOCATION
 19798  116504  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
 19799  116512  032737  000004  110152          BIT     #BIT02,RECDAT           ;LAST (R1) HIT?
 19800  116520  001402                          BEQ     5$                      ;IF NOT, QBUS MEMORY
 19801  116522  005202                          INC     R2                      ;INCREMENT 1K COUNT FOR PMI
 19802  116524  000401                          BR      6$                      ;GO CONITNUE
 19803  116526  005203                  5$:     INC     R3                      ;INCREMENT COUNT FOR QBUS MEM.
 19804  116530  022701  154000          6$:     CMP     #154000,R1              ;LAST IN 4K PAGE?
 19805  116534  101357                          BHI     3$                      ;IF NOT, BRANCH
 19806  116536  022737  177600  172354          CMP     #177600,KIPAR6          ;2M BOUNDARY?
 19807  116544  001345                          BNE     1$                      ;IF NOT, BRANCH
 19808  116546  000402                          BR      8$                      ;IF 2M, DON'T TOUCH STACK
 19809                                  ;
 19810                                  ; MAPPING IS DONE, FIND OUT HOW MANY PAGES WERE THERE
 19811                                  ;
 19812  116550  005726                  7$:     TST     (SP)+                   ;ADJUST STACK
 19813  116552  005726                          TST     (SP)+
 19814  116554  010437  000004          8$:     MOV     R4,ERRVEC               ;RESTORE ERROR VECTOR
 19815  116560  005037  177572                  CLR     MMR0                    ;DISABLE MEMORY MANGEMENT
 19816  116564  042737  000020  172516          BIC     #BIT04,MMR3             ;DISABLE 22 BITS
 19817  116572  005702                          TST     R2                      ;ANY PMI MEMORY?
 19818  116574  001405                          BEQ     9$                      ;IF NOT, GO CHECK QBUS MEMORY
 19819  116576  006302                          ASL     R2                      ;TRANSLATE TO BYTES
 19820  116600  010246                          MOV     R2,-(SP)                ;STORE 1K # ON STACK
 19821  116602  104405                          TYPDS                           ;TYPE # OF PAGES
 19822  116604  104401  116630                  TYPE    ,MEMK                   ;TYPE ASCII
 19823  116610  005703                  9$:     TST     R3                      ;ANY Q-BUS MEMORY?
 19824  116612  001405                          BEQ     10$                     ;IF NOT, BRANCH
 19825  116614  006303                          ASL     R3                      ;TRANSLATE TO BYTES
 19826  116616  010346                          MOV     R3,-(SP)                ;STORE 1K # ON STACK
 19827  116620  104405                          TYPDS                           ;TYPE # OF PAGES
 19828  116622  104401  116660                  TYPE    ,MEMQ                   ;TYPE ASCII
 19829  116626                          10$:
 19830  116626  000431                          BR      TST55           ;;EXIT TEST
 19831
 19832  116630  020113  054502  042524  MEMK:   .ASCIZ  /K BYTES OF PMI MEMORY/<12><15>
 19833  116636  020123  043117  050040
 19834  116644  044515  046440  046505
 19835  116652  051117  005131  000015
 19836  116660  020113  054502  042524  MEMQ:   .ASCIZ  /K BYTES OF Q-BUS MEMORY/<12><15>
 19837  116666  020123  043117  050440
 19838  116674  041055  051525  046440
```

```
 19839  116702  046505  051117  005131
 19840  116710  000015
 19841                                  .EVEN
 19842
 19843                                  .SBTTL WRONG PARITY ABORT TEST
 19844                                  ;WRONG PARITY ABORT
 19845                                  ;THIS TEST VERIFIES ABORT TO 114 USING PARITY OR ECC MEMORY CSR.
 19846                                  ;IF MORE THEN 1 CSR PRESENT, ALL OF THEM WILL BE WRITTEN AT THE
 19847                                  ;SAME TIME.
 19848                                  ;
 19849                                  ;ROUTINE TEST
 19850                                  ;.      SIZE FOR ALL POSSIBLE MEMORY CSR'S
 19851                                  ;.      STORE UP TO 16 CSR STARTING FROM TEMP
 19852                                  ;.      WRITE ALL WITH WRONG PARITY
 19853                                  ;.      WRITE TO 0
 19854                                  ;.      READ IT BACK
 19855                                  ;.      IF NO ABORT TO 114 THEN
 19856                                  ;.      .       ERROR IN PARITY ABORT LOGIC
 19857                                  ;.      ENDIF
 19858                                  ;.      RESTORE CSR'S
 19859                                  ;ENDROUTINE
 19860
 19861                                  ;;*****************************************************************
 19862  116712  000004                  TST55:  SCOPE
 19863                                  ;
 19864                                  ; FIND OUT ALL POSSIBLE MEMORY CSR LOCATIONS UP TO 16
 19865                                  ;
 19866  116714  013701  000004                  MOV     ERRVEC,R1                       ;STORE TIMEOUT VECTOR
 19867  116720  012737  116756  000004        MOV     #2$,ERRVEC                      ;POINT NEW TO PROGRAM
 19868  116726  012737  000340  000006        MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
 19869  116734  005004                        CLR     R4                              ;COUNT FOR CSR'S
 19870  116736  012702  172100                MOV     #172100,R2                      ;FIRST POSSIBLE CSR
 19871  116742  012703  002710                MOV     #TEMP,R3                        ;STORAGE LOCATION
 19872  116746  005712                1$:     TST     (R2)                            ;IS CSR THERE?
 19873  116750  010223                        MOV     R2,(R3)+                        ;IF THERE, STORE
 19874  116752  005204                        INC     R4                              ;INCREMENT COUNT FOR CSR'S
 19875  116754  000402                        BR      3$                              ;BRANCH AROUND
 19876  116756  005726                2$:     TST     (SP)+                           ;RESTORE STACK
 19877  116760  005726                        TST     (SP)+
 19878  116762  062702  000002        3$:     ADD     #2,R2                           ;POINT TO NEW CSR
 19879  116766  022702  172136                CMP     #172136,R2                      ;ALL DONE?
 19880  116772  101365                        BHI     1$                              ;IF NOT, BRANCH
 19881  116774  010137  000004                MOV     R1,ERRVEC                       ;RESTORE ERROR VECTOR
 19882  117000  052737  001000  177746        BIS     #BIT09,CCR                      ;SET CACHE BYPASS
 19883                                  ;
 19884                                  ; WRITE ALL CSR'S WITH WRONG ECC CODE
 19885                                  ; NOTE: IN PARITY MEMORY THOSE BITS ARE READ ONLY AND
 19886                                  ;       DIAGNOSTIC MODE BIT FOR ECC IS THE SAME AS WRONG PARITY
 19887                                  ;
 19888  117006  013701  000114                  MOV     @#114,R1                        ;STORE PARITY ABORT VECTOR
 19889  117012  012737  117100  000114        MOV     #6$,@#114                       ;POINT NEW TO PROGRAM
 19890  117020  012737  000340  000116        MOV     #340,@#116                      ;AT PRIORITY 7
 19891  117026  010402                        MOV     R4,R2                           ;STORE CSR COUNT
 19892  117030  012703  002710                MOV     #TEMP,R3                        ;START WITH 1ST CSR
 19893  117034  052773  000005  000000  4$:   BIS     #BIT02!BIT00,@0(R3)             ;DIAGNOSTIC OR WRONG PARITY
 19894  117042  042733  003740                BIC     #3740,@(R3)+                    ;CLEAR <10-5> CHECK BITS
```

```
19895  117046  077406                          SOB     R4,4$           ;DO FOR ALL CSR'S
19896  117050  005037  000000                  CLR     @#0             ;CLEAR TEST LOCATION WITH WRONG PR
19897  117054  010204                          MOV     R2,R4           ;RESTORE COUNTER
19898  117056  012703  002710                  MOV     #TEMP,R3        ;START WITH 1ST CSR
19899  117062  042733  000004          5$:     BIC     #BIT02,0(R3)+   ;CLEAR ALL WRONG PARITY
19900  117066  077203                          SOB     R2,5$           ;IN ALL CSR'S
19901  117070  005737  000000                  TST     @#0             ;READ BACK WRONG PARITY
19902  117074  104034                          ERROR   ^34             ;NO WRONG PARITY ABORT
19903  117076  000402                          BR      7$
19904  117100  005726                  6$:     TST     (SP)+           ;ADJUST STACK
19905  117102  005726                          TST     (SP)+
19906  117104  012703  002710          7$:     MOV     #TEMP,R3        ;START WITH 1ST CSR
19907  117110  042733  000001          8$:     BIC     #BIT00,0(R3)+   ;CLEAR ALL WRONG PARITY
19908  117114  077403                          SOB     R4,8$           ;IN ALL CSR'S
19909  117116  032737  100000  177744          BIT     #BIT15,MSER     ;ABORT REFLECTED IN MSER?
19910  117124  001004                          BNE     9$              ;IF YES, BRANCH
19911  117126  012737  100000  001124          MOV     #100000,$GDDAT  ;
19912  117134  104035                          ERROR   ^35             ;MSER<15> NOT SET ON PARITY ABORT
19913  117136  042737  001000  177746  9$:     BIC     #BIT09,CCR      ;CLEAR CACHE BYPASS
19914  117144  005037  177744                  CLR     MSER            ;CLEAR MSER
19915  117150  010137  000114                  MOV     R1,@#114        ;RESTORE PARITY ABORT
19916
19917
```

```
                                .SBTTL  TEST  - DMA TAG PARITY IN STANDALONE MODE
                                ;CHECK DMA TAG STORE PARITY BIT.
                                ;ROUTINE TEST
                                ;.      CACHE DMA_PARITY
                                ;.      LET BCSR<08>=#1
                                ;.      REPORT ALL ERRORS
                                ;ENDROUTINE
                                ;
                                ;ROUTINE DMA_PARITY
                                ;.      GENERATE PARITY ERRORS
                                ;.      CHECK MSER<13>
                                ;.      LET BCSR<08>=#0
                                ;ENDROUTINE

                                ;;*****************************************************************
117154  000004                  TST56:  SCOPE
                                ;
                                ; ALLOCATE CODE IN CACHE
                                ;
117156  012702  117204                  MOV     #DMAPAR,R2                      ;POINT TO STANDALONE CODE
117162  005722                  1$:     TST     (R2)+                           ;ALLOCATE IN CACHE
117164  022702  117244                  CMP     #DPAREN,R2                      ;LAST ADDRESS?
117170  001374                          BNE     1$                              ;IF NOT, BRANCH
117172  005737  002710                  TST     TEMP                            ;ALLOCATE TEST LOCATION
117176  013737  177520  002710          MOV     BCSR,SAVBR                      ;SAVE BCSR
                                ;
                                ; IN STANDALONE MODE TRY TO VERIFY RESPONSE TO PARITY ERRORS
                                ;
117204  052737  000400  177520  DMAPAR: BIS     #BIT08,BCSR                     ;SET STANDALONE MODE BIT
117212  052737  002001  177746          BIS     #BIT10!BIT00,CCR                ;WRITE WRONG TAG PARITY
117220  005037  002710                  CLR     TEMP                            ;WRITE HIT WITH WRONG PARITY
117224  013705  177744                  MOV     MSER,R5                         ;STORE MSER
117230  042737  002000  177746          BIC     #BIT10,CCR                      ;CLEAR WRONG PARITY BIT
117236  042737  000400  177520  2$:     BIC     #BIT08,BCSR                     ;CLEAR STANDALONE BIT
                                ;
                                ; RETURN FROM STANDALONE MODE
                                ;
117244                          DPAREN:
117244  022705  060020                  CMP     #60020,R5                       ;WRONG DMA PARITY?
117250  001401                          BEQ     4$                              ;IF OK, BRANCH
117252  104117                          ERROR   +117                            ;MSER NOT SET IN STANDALONE MODE
117254  005037  177744          4$:     CLR     MSER
117260  012737  000400  177746          MOV     #400,CCR                        ;FLUSH THE CACHE
117266  013737  002710  177520          MOV     SAVBR,BCSR                      ;RESTORE BCSR
```

```
 19963                                  .SBTTL  TEST  - DMA TAG PARITY W/O STANDALONE MODE
 19964                                  ;CHECK DMA TAG PARITY BIT WITHOUT GOING INTO STANDALONE MODE.
 19965                                  ;
 19966                                  ;CCR    <10>    WRITE WRONG TAG PARITY
 19967                                  ;
 19968                                  ;ROUTINE TEST
 19969                                  ;.      LET CCR<10>=#1
 19970                                  ;.      ALLOCATE LOCATION IN CACHE
 19971                                  ;.      INITIATE DMA WRITE TRANSFERS
 19972                                  ;.      IF MSER<04> NE #1 THEN
 19973                                  ;.          ERROR
 19974                                  ;.      ENDIF
 19975                                  ;ENDROUTINE
 19976                                  ;
 19977                                  ;;*****************************************************************
 19978  117274  000004                  TST57:  SCOPE
 19979  117276  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
 19980  117304  001402                          BEQ     110$                            ;IF NOT, BRANCH
 19981  117306  000137  132724                  JMP     $EOP                            ;OTHERWISE, NEXT PASS
 19982  117312  005737  002644          110$:   TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
 19983  117316  001457                          BEQ     TST60           ;;IF NOT, EXIT TEST
 19984                                  ;
 19985                                  ; ALLOCATE TEST IN CACHE
 19986                                  ;
 19987  117320  012703  117320          11$:    MOV     #.,R3                           ;START WITH CURRENT INSTRUCTION
 19988  117324  005723                  10$:    TST     (R3)+                           ;READ A WORD
 19989  117326  022703  117436                  CMP     #4$,R3                          ;ALL DONE?
 19990  117332  001374                          BNE     10$                             ;IF NOT, CONTINUE
 19991                                  ;
 19992                                  ; WRITE A WORD WITH WRONG PARITY
 19993                                  ;
 19994  117334  013737  000114  001160  1$:     MOV     @#114,$TMP0                     ;STORE PARITY VECTOR
 19995  117342  012737  117410  000114          MOV     #2$,@#114                       ;POINT TO TEST AREA
 19996  117350  052737  002000  177746          BIS     #BIT10,CCR                      ;WRITE WRONG TAG PARITY
 19997  117356  005037  002710                  CLR     TEMP                            ;WRITE MISS WITH WRONG TAG PARITY
 19998  117362  042737  002000  177746          BIC     #BIT10,CCR                      ;CLEAR WRONG TAG PARITY
 19999  117370  052737  000200  177746          BIS     #BIT07,CCR                      ;PARITY ABORT
 20000  117376  005000                          CLR     R0                              ;FLAG TO DO 1 TRANSFER
 20001  117400  004737  132162                  JSR     PC,DMATRN                       ;DO DMA WRITE TO TEMP THRU Q22BE
 20002  117404  104116                          ERROR   .116                            ;NO PARITY ABORT
 20003  117406  000413                          BR      4$                              ;BRANCH TO TEST MSER
 20004  117410  013704  177744          2$:     MOV     MSER,R4                         ;STORE REGISTER
 20005  117414  005037  177744                  CLR     MSER                            ;CLEAR MSER
 20006  117420  005726                          TST     (SP)+                           ;
 20007  117422  005726                          TST     (SP)+                           ;RESTORE STACK
 20008  117424  005726                          TST     (SP)+
 20009  117426  032704  000020          3$:     BIT     #BIT04,R4                       ;MSER OK?
 20010  117432  001001                          BNE     4$                              ;IF SET, BRANCH
 20011  117434  104116                          ERROR   .116                            ;MSER<4> NOT SET
 20012  117436  005037  177744          4$:     CLR     MSER                            ;CLEAR MSER
 20013  117442  012737  000400  177746          MOV     #400,CCR                        ;FLUSH CACHE
 20014  117450  013737  001160  000114          MOV     $TMP0,@#114                     ;RESTORE VECTOR
```

```
20015                                   .SBTTL  TEST  - DMA WRITE HIT CYCLES
20016                                   ;CHECK THAT DMA WRITE HITS INVALIDATE CACHE.
20017                                   ;ROUTINE TEST
20018                                   ;.      ALLOCATE A LOCATION IN CACHE
20019                                   ;.      INITIATE DMA WRITE TO THIS LOCATION
20020                                   ;.      READ THIS LOCATION BACK
20021                                   ;.      IF IT IS CHANGED OR HIT/MISS EQ HIT
20022                                   ;.      .       ERROR
20023                                   ;.      ENDIF
20024                                   ;.      WRITE TO THE FIRST 16 LOCATION THEIR ADDRESS
20025                                   ;.      DO BLOCK MODE TRANSFER TO THOSE LOCATIONS
20026                                   ;.      START READ WITH THE LAST LOCATION
20027                                   ;.      IF RECORD ANY HITS THEN
20028                                   ;.      .       ERROR
20029                                   ;.      ENDIF
20030                                   ;.      INITIATE READ DMA TO THE SAME TEST LOCATIONS
20031                                   ;.      READ THEM BACK
20032                                   ;.      IF HIT/MISS NE HIT THEN
20033                                   ;.      .       ERROR
20034                                   ;.      ENDIF
20035                                   ;ENDROUTINE
20036
20037                                   ;;*****************************************************************
20038  117456  000004                  TST60:  SCOPE
20039  117460  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
20040  117466  001402                          BEQ     1$                              ;IF NOT, BRANCH
20041  117470  000137  132724                  JMP     $EOP                            ;OTHERWISE, NEXT PASS
20042  117474  005737  002644          1$:     TST     CSR1                            ;AT LEAST 1 Q22BE?
20043  117500  001002                          BNE     2$                              ;IF YES, BRANCH
20044  117502  000137  132724                  JMP     $EOP                            ;OTHERWISE, NEXT PASS
20045                                   ;
20046                                   ; TRY TO DO DMA WRITE AT ALL DIFFERENT PRIORITIES
20047                                   ;
20048  117506  012702  000340          2$:     MOV     #340,R2                         ;START WITH 7
20049  117512  000402                          BR      4$                              ;GO DO IT
20050  117514  162702  000040          3$:     SUB     #40,R2                          ;LOWER PRIORITY
20051  117520  005037  002710          4$:     CLR     TEMP                            ;CLEAR TEST LOCATION
20052  117524  005000                          CLR     R0                              ;DO JUST 1 WORD
20053  117526  106402                          MTPS    R2                              ;LOWER PRIORITY
20054  117530  004737  132162                  JSR     PC,DMATRN                       ;DO DATO
20055  117534  005737  002710                  TST     TEMP                            ;STILL CACHED?
20056  117540  013737  177752  110152          MOV     HITMIS,RECDAT                   ;STORE HIT/MISS
20057  117546  032737  000004  110152          BIT     #BIT02,RECDAT                   ;LAST ACCESS HIT?
20058  117554  001401                          BEQ     5$                              ;IF MISS, BRANCH
20059  117556  104120                          ERROR   .120
20060  117560  022737  012525  002710  5$:     CMP     #12525,TEMP                     ;DATO OK?
20061  117566  001401                          BEQ     6$                              ;IF SO, BRANCH
20062  117570  104123                          ERROR   .123                            ;DATO
20063  117572  005702                  6$:     TST     R2                              ;LAST PRIORITY 0?
20064  117574  001347                          BNE     3$                              ;IF NOT, BRANCH
20065  117576  106427  000340                  MTPS    #340                            ;RESTORE PRIORITY
20066                                   ;
20067                                   ; VERIFY THAT DMA WITH FORCE MISS DOES NOT INVALIDATE CACHE
20068                                   ;
20069  117602  005037  002710                  CLR     TEMP                            ;ALLOCATE CACHE
20070  117606  012737  000004  177746          MOV     #BIT2,CCR                       ;SET FORCE MISS
```

```
 20071  117614  004737  132162                  JSR     PC,DMATRN                       ;DO DMA DATI
 20072  117620  005037  177746                  CLR     CCR                             ;CLEAR FORCE MISS
 20073  117624  005737  002710                  TST     TEMP                            ;STILL IN CACHE
 20074  117630  013737  177752  110152          MOV     HITMIS,RECDAT                   ;STORE REGISTER
 20075  117636  032737  000004  110152          BIT     #BIT02,RECDAT                   ;HIT?
 20076  117644  001001                          BNE     7$                              ;IF SO, BRANCH
 20077  117646  104123                          ERROR   .123
 20078  117650  005737  002710          7$:     TST     TEMP                            ;READ FROM CACHE?
 20079  117654  001401                          BEQ     8$                              ;IF STILL 0, BRANCH
 20080  117656  104123                          ERROR   .123
 20081                                  ;
 20082                                  ; VERIFY THAT BYPASS WITH DMA DATI INVALIDATES CACHE
 20083                                  ;
 20084  117660  012737  001000  177746  8$:     MOV     #BIT09,CCR                      ;SET BYPASS
 20085  117666  004737  132162                  JSR     PC,DMATRN                       ;DO DMA DATI
 20086  117672  005037  177746                  CLR     CCR                             ;CLEAR BYPASS
 20087  117676  005737  002710                  TST     TEMP                            ;IN CACHE?
 20088  117702  013737  177752  110152          MOV     HITMIS,RECDAT                   ;STORE REGISTER
 20089  117710  032737  000004  110152          BIT     #BIT02,RECDAT                   ;TEMP WAS A HIT?
 20090  117716  001401                          BEQ     DATBO                           ;IF NOT, BRANCH
 20091  117720  104123                          ERROR   .123
 20092                                  ;
 20093                                  ; DO DATBO
 20094                                  ;
 20095  117722  012701  000010          DATBO:  MOV     #10,R1                          ;COUNTER FOR 8
 20096  117726  012702  002710                  MOV     #TEMP,R2                        ;START WITH TEMP
 20097  117732  005022                  1$:     CLR     (R2)+                           ;CLEAR ALL 16
 20098  117734  077102                          SOB     R1,1$                           ;DO ALL 16
 20099  117736  005200                          INC     R0                              ;FLAG BLOCK MODE
 20100  117740  012777  002710  062702          MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
 20101  117746  012777  177770  062676          MOV     #177770,@WC                     ;DO 8 WORDS
 20102  117754  012777  001701  062662          MOV     #1701,@CSR1                     ;16 WORDS FROM 32K
 20103  117762  004737  132162                  JSR     PC,DMATRN                       ;DO DATBO
 20104  117766  032777  010000  062652          BIT     #BIT12,@CSR2                    ;NO BLOCK MODE SLAVE?
 20105  117774  001401                          BEQ     2$                              ;IF NOT, BRANCH
 20106  117776  104123                          ERROR   .123                            ;NO BLOCK MODE SLAVE
 20107  120000  012701  000004          2$:     MOV     #4,R1                           ;COUNTER FOR 4 LOCATIONS
 20108  120004  012702  002710                  MOV     #TEMP,R2                        ;START WITH TEMP
 20109  120010  005712                  3$:     TST     (R2)                            ;ACCESS A LOCATION
 20110  120012  013737  177752  110152          MOV     HITMIS,RECDAT                   ;STORE REGISTER
 20111  120020  032737  000004  110152          BIT     #BIT02,RECDAT                   ;HIT?
 20112  120026  001401                          BEQ     4$                              ;IF NOT, BRANCH
 20113  120030  104121                          ERROR   .121                            ;DMA DOES NOT INVALIDATE
 20114  120032  022722  012525          4$:     CMP     #12525,(R2)+                    ;DATO OK?
 20115  120036  001401                          BEQ     5$                              ;IF SO,BRANCH
 20116  120040  104123                          ERROR   .123                            ;IN DMA
 20117  120042  062702  000002          5$:     ADD     #2,R2                           ;DO IN 2 WORDS
 20118  120046  077120                          SOB     R1,3$                           ;DO ALL 16
 20119
 20120                                  ;
 20121                                  ; DO 4K OF DATO AND CHECK FOR MISS
 20122                                  ;
 20123  120050  004737  131362                  JSR     PC,INITMM                       ;SET UP MMU
 20124  120054  012737  002000  172354          MOV     #2000,KIPAR6                    ;START AT 32K
 20125  120062  042737  100000  172314          BIC     #BIT15,KIPDR6                   ;NO BYPASS
 20126  120070  052737  000001  177572          BIS     #BIT00,MMR0                     ;ENABLE MMU
```

```
 20127  120076  012737  120222  000004          MOV     #DATI,@#4               ;IF 32K NXM
 20128  120104  012702  140000                  MOV     #140000,R2              ;START WITH 32K
 20129  120110  005712                          TST     (R2)                    ;EXIT
 20130  120112  012701  010000                  MOV     #10000,R1               ;COUNTER FOR 4K
 20131  120116  005022                  6$:     CLR     (R2)+                   ;CLEAR ALL 16
 20132  120120  077102                          SOB     R1,6$                   ;DO ALL 16
 20133  120122  005200                          INC     R0                      ;FLAG BLOCK MODE
 20134  120124  012777  000000  062516          MOV     #0,@BBA                 ;LOAD DMA ADDRESS
 20135  120132  012777  170000  062512          MOV     #-10000,@BWC            ;DO 4K
 20136  120140  012777  003701  062476          MOV     #3701,@BCSR1            ;16 WORDS FROM 32K
 20137  120146  004737  132162                  JSR     PC,DMATRN               ;DO DATBO
 20138  120152  012701  004000          7$:     MOV     #4000,R1                ;COUNTER FOR 4K LOCATIONS
 20139  120156  012702  140000                  MOV     #140000,R2              ;START WITH 0
 20140  120162  005712                  8$:     TST     (R2)                    ;ACCESS A LOCATION
 20141  120164  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
 20142  120172  032737  000004  110152          BIT     #BIT02,RECDAT           ;HIT?
 20143  120200  001401                          BEQ     9$                      ;IF NOT, BRANCH
 20144  120202  104121                          ERROR   •121                    ;DMA DOES NOT INVALIDATE
 20145  120204  022722  012525          9$:     CMP     #12525,(R2)+            ;DATO OK?
 20146  120210  001401                          BEQ     10$                     ;IF SO,BRANCH
 20147  120212  104123                          ERROR   •123                    ;IN DMA
 20148  120214  062702  000002          10$:    ADD     #2,R2                   ;DO IN 2 WORDS
 20149  120220  077120                          SOB     R1,8$                   ;DO ALL OF THEM
 20150                                  ;
 20151                                  ; CHECK DATI
 20152                                  ;
 20153  120222  005037  177572          DATI:   CLR     @MMR0                   ;DISABLE MMU
 20154  120226  012706  001100                  MOV     #1100,SP                ;RESTORE STACK
 20155  120232  012737  132334  000004          MOV     #TOUT,@#4               ;AND TIMEOUT
 20156  120240  012737  052525  002710          MOV     #52525,TEMP             ;LOAD MEMORY
 20157  120246  005000                          CLR     R0                      ;JUST 1 WORD
 20158  120250  004737  132244                  JSR     PC,DMARD                ;DO DATI
 20159  120254  022777  052525  062372          CMP     #52525,@DATA            ;DATI OK?
 20160  120262  001401                          BEQ     11$                     ;IF YES, BRANCH
 20161  120264  104123                          ERROR   •123                    ;DATI
 20162                                  ;
 20163                                  ; DO DATBI
 20164                                  ;
 20165  120266  012701  000010          11$:    MOV     #10,R1                  ;COUNTER FOR 16 LOCATIONS
 20166  120272  012702  002710                  MOV     #TEMP,R2                ;START WITH TEMP
 20167  120276  012703  002710                  MOV     #TEMP,R3                ;START WITH TEMP
 20168  120302  010322          12$:    MOV     R3,(R2)+                ;PUT ADDRESSES
 20169  120304  005723                          TST     (R3)+                   ;INCREMENT ADDRESS
 20170  120306  077103                          SOB     R1,12$                  ;DO ALL 16
 20171  120310  005200                          INC     R0                      ;FLAG BLOCK MODE
 20172  120312  004737  132244                  JSR     PC,DMARD                ;DO DATBI
 20173  120316  032777  010000  062322          BIT     #BIT12,@CSR2            ;NO BLOCK MODE SLAVE?
 20174  120324  001401                          BEQ     13$                     ;IF NO, BRANCH
 20175  120326  104123                          ERROR   •123
 20176  120330  022777  002726  062316  13$:    CMP     #TEMP+16,@DATA          ;DATI OK?
 20177  120336  001401                          BEQ     14$                     ;IF SO, BRANCH
 20178  120340  104123                          ERROR   •123                    ;IN DATIB
 20179  120342  005737  002710          14$:    TST     TEMP                    ;ACCESS
 20180  120346  013737  177752  110152          MOV     HITMIS,RECDAT           ;STORE REGISTER
 20181  120354  032737  000004  110152          BIT     #BIT02,RECDAT           ;HIT?
 20182  120362  001001                          BNE     15$                     ;IF YES, BRANCH
```

```
 20183  120364  104123                          ERROR   +123
 20184  120366                         15$:
 20185
```

TEST - DIFFERENT LEVELS OF INTERRUPTS SEQ 0387

```
20186                                   .SBTTL  TEST  - DIFFERENT LEVELS OF INTERRUPTS
20187                                   ;DIFFERENT LEVELS OF INTERRUPTS
20188                                   ;THIS TEST WILL PROGRAM Q22 BUS EXERCISER TO INTERRUPT AT DIFFERENT
20189                                   ;LEVELS.  ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS AND
20190                                   ;PIRQ'S WILL BE TESTED.
20191                                   ;
20192                                   ;CHECK DIFFERENT LEVELS OF INTERRUPTS.
20193                                   ;ROUTINE TEST
20194                                   :.      SET VECTOR TO INTERRUPT_DMA
20195                                   :.      FOR INTERRUPTS FROM 4 TO 7 DO
20196                                   :.      .       ENABLE INTERRUPTS
20197                                   :.      .       SET PRIORITY=INTERUPT
20198                                   :.      .       IF INTERRUPT_FLAG SET THEN
20199                                   :.      .       .       ERROR
20200                                   :.      .       ENDIF
20201                                   :.      .       ENABLE INTERRUPTS
20202                                   :.      .       SET PRIORITY=INTERRUPT-1
20203                                   :.      .       IF INTERRUPT_FLAG NOTSET THEN
20204                                   :.      .       .       ERROR
20205                                   :.      .       ENDIF
20206                                   :.      .       LET INTERRUPT_DMA=0
20207                                   :.      ENDDO
20208                                   ;ENDROUTINE
20209                                   ;
20210                                   ;ROUTINE INTERUPT_DMA
20211                                   :.      LET INTERRUPT_FLAG=1
20212                                   ;RETURN
20213                                   ;ENDROUTINE
20214
20215                                   ;;*****************************************************************
20216   120366  000004                  TST61:  SCOPE
20217   120370  032737  000040  000052          BIT     #BIT05,@#52                     ;UFD MODE?
20218   120376  001122                          BNE     TST62           ;;IF SO, EXIT TEST
20219   120400  005737  002644                  TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
20220   120404  001517                          BEQ     TST62           ;;IF NOT, EXIT TEST
20221                                   ;
20222                                   ; SETUP INITIAL PRIORITY TO 7
20223                                   ;
20224   120406  013703  002644                  MOV     CSR1,R3                         ;DO FOR FIRST FOUND Q22BE
20225   120412  012777  120506  062236          MOV     #5$,@VQBE1                      ;POINT INTERRUPT VECTOR TO PROGRAM
20226   120420  012777  000340  062232          MOV     #340,@VQPR1                     ;AT PRIORITY 7
20227   120426  012700  002752                  MOV     #Q22EN,R0                       ;START WITH 7 FOR INTERRUPTS
20228   120432  012701  000340                  MOV     #340,R1                         ;LOW BOUNDARY FOR NO INTERRUPTS
20229   120436  000402                          BR      2$                              ;TRY TO DO IT FOR FIRST
20230   120440  162701  000040          1$:     SUB     #40,R1                          ;LOWER LOW BOUNDARY
20231                                   ;
20232                                   ; CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN BR
20233                                   ;
20234   120444  012737  000340  001160  2$:     MOV     #340,$TMP0                      ;TOP PRIORITY FOR NO INTERRUPTS
20235   120452  000403                          BR      4$                              ;DO FIRST ONE
20236   120454  162737  000040  001160  3$:     SUB     #40,$TMP0                       ;DO AT NEXT LEVEL
20237   120462  106437  001160          4$:     MTPS    $TMP0                           ;SET PRIORITY NOT TO INTERRUPT
20238   120466  004737  132142                  JSR     PC,Q22INT                       ;ENABLE INTERRUPTS
20239   120472  012077  062150                  MOV     (R0)+,@CSR2                     ;CLEAR GO BIT
20240   120476  000240                          NOP
20241   120500  000240                          NOP
```

```
20242  120502  000240                          NOP
20243  120504  000403                          BR      6$                      ;IF NO INTERUPT, BRANCH
20244  120506  104126                  5$:     ERROR   +126                    ;INTERRUPTS HAPPEN
20245  120510  005726                          TST     (SP)+                   ;RESTORE STACK
20246  120512  005726                          TST     (SP)+
20247  120514  020137  001160          6$:     CMP     R1,$TMP0                ;LAST ONE?
20248  120520  001355                          BNE     3$                      ;IF NOT BRANCH
20249  120522  022710  000002                  CMP     #2,(R0)                 ;AT BR4?
20250  120526  001344                          BNE     1$                      ;IF NOT LAST ONE, BRANCH
20251                                  ;
20252                                  ; INTERRUPT AT ALL LEVELS
20253                                  ;
20254  120530  012777  120624  062120  INQ22:  MOV     #5$,@VQBE1              ;POINT INTERRUPT VECTOR TO PROGRAM
20255  120536  012777  000340  062114          MOV     #340,@VQPR1             ;AT PRIORITY 7
20256  120544  012700  002752                  MOV     #Q22EN,R0               ;START WITH 7 FOR INTERRUPTS
20257  120550  012701  000300                  MOV     #300,R1                 ;TOP BOUNDARY FOR INTERRUPTS
20258  120554  000402                          BR      2$                      ;TRY TO DO IT FOR FIRST
20259  120556  162701  000040          1$:     SUB     #40,R1                  ;LOWER TOP BOUNDARY
20260                                  ;
20261                                  ; CHECK THAT INTERRUPTS HAPPEN AT PRIORITY LOWER THAN BR
20262                                  ;
20263  120562  010137  001160          2$:     MOV     R1,$TMP0                ;PRIORITY FOR INTERRUPTS
20264  120566  000403                          BR      4$                      ;DO FIRST ONE
20265  120570  162737  000040  001160  3$:     SUB     #40,$TMP0               ;DO AT NEXT LEVEL
20266  120576  106437  001160          4$:     MTPS    $TMP0                   ;SET PRIORITY NOT TO INTERRUPT
20267  120602  004737  132142                  JSR     PC,Q22INT               ;ENABLE INTERRUPTS
20268  120606  011077  062034                  MOV     (R0),@CSR2              ;CLEAR GO BIT
20269  120612  000240                          NOP                             ;WAIT A WHILE
20270  120614  000240                          NOP
20271  120616  000240                          NOP
20272  120620  104126                          ERROR   +126                    ;INTERRUPTS DON'T HAPPEN
20273  120622  000402                          BR      6$                      ;DON'T RESTORE STACK
20274  120624  005726                  5$:     TST     (SP)+                   ;RESTORE STACK
20275  120626  005726                          TST     (SP)+
20276  120630  005737  001160          6$:     TST     $TMP0                   ;LAST ONE 0?
20277  120634  001355                          BNE     3$                      ;IF NOT BRANCH
20278  120636  022720  000002                  CMP     #2,(R0)+                ;AT BR4?
20279  120642  001345                          BNE     1$                      ;IF NOT LAST ONE, BRANCH
20280
20281
```

```
                                       .SBTTL  TEST  - ARBITRATION BETWEEN PIRQ'S AND INTERRUPTS
                                       ;CHECK PRIORITY ORDER BETWEEN PIRQ'S AND INTERRUPTS.
                                       ;ROUTINE TEST
                                       ;.      IF UFD THEN
                                       ;.      .       EXIT TEST
                                       ;.      ENDIF
                                       ;.      DO FOR I FROM #6 DOWN TO #3
                                       ;.      .       SET PRIORITY TO I
                                       ;.      .       ENABLE INTERRUPT(I+1) AND PIRQ(I+1)
                                       ;.      .       IF INTERRUPT(I+1) WAS BEFORE PIRQ(I+1) THEN
                                       ;.      .       .       ERROR
                                       ;.      .       ENDIF
                                       ;.      ENDDO
                                       ;ENDROUTINE

                                       ;;**************************************************************
120644  000004                 TST62:  SCOPE
120646  032737  000040  000052         BIT     #BIT05,@#52                     ;UFD MODE?
120654  001065                       BNE     TST63           ;;IF SO, EXIT TEST
120656  005737  002644               TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
120662  001462                       BEQ     TST63           ;;IF NOT, EXIT TEST
120664  012777  120772  061764       MOV     #3$,@VQBE1                      ;SETUP Q22BE VECTOR
120672  012777  000340  061760       MOV     #340,@VQPR1                     ;AT PRIORITY 7
120700  012737  121002  000240       MOV     #4$,PIRQVEC                     ;SETUP PIRQ VECTOR
120706  012737  000340  000242       MOV     #340,PIRQVEC+2                  ;AT PRIORITY 7
120714  012700  002752               MOV     #Q22EN,R0                       ;POINT THRU PRIORITIES FOR Q22BE
120720  012704  121020               MOV     #PIRQT,R4                       ;POINTER THRU PIRQ'S
120724  013703  002644               MOV     CSR1,R3                         ;DO FOR FIRST Q22BE
120730  012702  000300               MOV     #300,R2                         ;START WITH CPU PRIORITY AT 7
120734  000402                       BR      2$                              ;DO FIRST ONE
120736  162702  000040       1$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
120742  106427  000340       2$:     MTPS    #340                            ;RAISE PRIORITY TO 7
120746  012437  177772               MOV     (R4)+,PIRQ                      ;SET PRIORITY FOR PIRQ'S
120752  004737  132142               JSR     PC,Q22INT                       ;INITIALISE Q22BE TO INTERRUPT
120756  012077  061664               MOV     (R0)+,@CSR2                     ;SET DONE BIT
120762  106402                       MTPS    R2                              ;LOWER PRIORITY
120764  000240                       NOP
120766  000240                       NOP
120770  000240                       NOP
120772  104124               3$:     ERROR   +124                            ;PIRQ'S DON'T TAKE OVER BIRQ'S
120774  005726                       TST     (SP)+                           ;CLEAN UP STACK
120776  005726                       TST     (SP)+
121000  000402                       BR      5$                              ;BRANCH AROUND PIRQ INTERRUPT
121002  005726               4$:     TST     (SP)+                           ;CLEAN UP STACK
121004  005726                       TST     (SP)+
121006  022702  000140       5$:     CMP     #140,R2                         ;PRIORITY 3 LAST ONE?
121012  001351                       BNE     1$                              ;IF NOT BRANCH
121014  005037  177772               CLR     PIRQ                            ;CLEAR ANY REQUESTS

121020  100000  040000  020000  PIRQT: .WORD  100000,40000,20000,10000        ;PIRQ'S<7-4>
121026  010000
```

```
                                        .SBTTL  TEST      POWER DOWN TEST
                                        ;USING Q22BE THIS TEST WILL CHECK THAT ON POWER DOWN CONDITION IF
                                        ;POWER UP CODE 00 IS SELECTED THE CPU TRAPS THRU 24
                                        ;ROUTINE TEST
                                        ;.      IF UFD OR POWER UP CODE 00 NOT SELECTED THEN
                                        ;.      .       EXIT TEST
                                        ;.      ENDIF
                                        ;.      SET 24 TO POINT TO TEST AREA
                                        ;.      LET CSR2<5> = #1 TO NEGATE BPOK
                                        ;.      IF NO TRAP TO 24 THEN
                                        ;.      .       ERROR IN POWER DOWN CYCLE
                                        ;.      ENDIF
                                        ;.      IF TRAP TO 24 THEN
                                        ;.      .       LET CSR2<5> = #0
                                        ;.      ENDIF
                                        ;ENDROUTINE

                                        ;;**************************************************************
121030  000004                  TST63:  SCOPE
121032  032737  000040  000052          BIT     #BIT05,@#52                             ;UFD MODE?
121040  001033                          BNE     TST64           ;;EXIT TEST IN UFD  MODE
121042  005737  002644                  TST     CSR1                                    ;AT LEAST ONE Q22BE FOUND?
121046  001430                          BEQ     TST64           ;;IF NOT, EXIT TEST
121050  013737  000024  001160          MOV     PWRVEC,$TMP0                            ;SAVE POWER UP VECTOR
121056  012737  121112  000024          MOV     #1$,PWRVEC                              ;POINT NEW TO PROGRAM
121064  012737  000340  000026          MOV     #340,PWRVEC+2                           ;AT PRIORITY 7
121072  012777  000040  061546          MOV     #BIT05,@CSR2                            ;DO POWER DOWN
121100  000240                          NOP
121102  005077  061540                  CLR     @CSR2                                   ;CLEAR POWER DOWN BIT
121106  104125                          ERROR   +125                                    ;NO POWER DOWN TRAP
121110  000404                          BR      2$                                      ;SKIP RESTORING STACK
121112  005077  061530          1$:     CLR     @CSR2                                   ;CLEAR POWER DOWN BIT
121116  005726                          TST     (SP)+                                   ;RESTORE STACK POINTER
121120  005726                          TST     (SP)+
121122  013737  001160  000024  2$:     MOV     $TMP0,PWRVEC                            ;RESTORE POWER VECTOR
```

```
                                TEST    - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS                    SEQ 0391

 20372                          .SBTTL TEST    - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS
 20373                          ;IF TWO Q22BE ARE AVAILABLE, THIS TEST WILL CHECK THAT HIGHER LEVEL
 20374                          ;INTERRUPT REQUESTS TAKE PRIORITY OVER LOWER LEVEL ONES
 20375                          ;ROUTINE TEST
 20376                          ;.      IF UFD OR 2ND Q22BE NOT AVAILABLE THEN
 20377                          ;.      .       EXIT TEST
 20378                          ;.      ENDIF
 20379                          ;.      DO FOR PRIORITY_LEVELS FROM 4 TO 6
 20380                          ;.      .       SET UP 1ST Q22BE TO INTERRUPT AT PRIORITY_LEVEL
 20381                          ;.      .       SET UP 2ND Q22BE TO INTERRUPT AT PRIORITY_LEVEL+1
 20382                          ;.      .       SET UP CPU PRIORITY TO PRIORITY_LEVEL-1
 20383                          ;.      .       IF INTERRUPTS FORM 1ST Q22BE HAPPENED BEFORE 1ST THEN
 20384                          ;.      .       .       ERROR
 20385                          ;.      .       ENDIF
 20386                          ;.      ENDDO
 20387                          ;ENDROUTINE
 20388
 20389                          ;;*****************************************************************
 20390  121130  000004          TST64:  SCOPE
 20391  121132  032737  000040  000052  BIT     @#BIT05,@#52                    ;UFD MODE?
 20392  121140  001064                  BNE     TST65           ;;IF SO, EXIT TEST
 20393  121142  005737  002664          TST     CSR12                           ;SECOND Q22BE AVAILABLE?
 20394  121146  001002                  BNE     1$                              ;IF YES, GO DO TEST
 20395  121150  000137  132724          JMP     $EOP                            ;OTHERWISE, GOTO EOP
 20396                          ;
 20397                          ; INITIALISE VECTORS FOR Q22BE'S
 20398                          ;
 20399  121154  012777  121270  061474  1$:     MOV     #4$,@VQBE1              ;VECTOR FOR 1ST ONE
 20400  121162  012777  000340  061470          MOV     #340,@VQPR1             ;AT PRIORITY 7
 20401  121170  012777  121300  061500          MOV     #5$,@VQBE2              ;VECTOR FOR 2ND ONE
 20402  121176  012777  000340  061474          MOV     #340,@VQPR2             ;AT PRIORITY 7
 20403  121204  012700  002754                  MOV     #Q22EN+2,R0             ;POINTER FOR Q22BE BR'S
 20404  121210  012702  000240                  MOV     #240,R2                 ;CPU AT 5
 20405  121214  000402                  BR      3$                              ;START DOING ARBITRATION
 20406                          ;
 20407                          ; DO FOR CPU PRIORITIES 5-3
 20408                          ;
 20409  121216  162702  000040  2$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
 20410  121222  106427  000340  3$:     MTPS    #340                            ;CPU AT 7
 20411  121226  013703  002664          MOV     CSR12,R3                        ;Q22BE 2 AT HIGHER BR
 20412  121232  004737  132142          JSR     PC,Q22INT                       ;INITIALISE TO INTERRUPTS
 20413  121236  011077  061404          MOV     (R0),@CSR2                      ;START 1ST ONE
 20414  121242  013703  002644          MOV     CSR1,R3                         ;Q22BE 1 AT LOWER BR
 20415  121246  005740                  TST     -(R0)                           ;HIGHER PRIORITY
 20416  121250  004737  132142          JSR     PC,Q22INT                       ;INITIALISE
 20417  121254  012077  061406          MOV     (R0)+,@CSR22                    ;START 2ND ONE
 20418  121260  106402                  MTPS    R2                              ;LOWER CPU PRIORITY
 20419  121262  000240                  NOP                                     ;WAIT A WHILE
 20420  121264  000240                  NOP
 20421  121266  000240                  NOP
 20422  121270  104126          4$:     ERROR   +126                            ;INTERRUPTS IN WRONG ORDER
 20423  121272  005726                  TST     (SP)+                           ;RESTORE STACK
 20424  121274  005726                  TST     (SP)+
 20425  121276  000402                  BR      6$
 20426  121300  005726          5$:     TST     (SP)+                           ;RESTORE STACK
 20427  121302  005726                  TST     (SP)+
```

```
 20428  121304  022702  000140          6$:     CMP     #140,R2                         ;PRIORITY 3 LAST?
 20429  121310  001342                          BNE     2$                              ;IF NOT, BRANCH TO CONTINUE
 20430
```

```
                                        .SBTTL  TEST   - PMG COUNTER
                                        ;USING 2 Q22BE THIS TEST WILL VALIDATE THAT PMG COUNTER IS REALLY
                                        ;CAPABLE OF GRANTING CPU BUS MASTERSHIP WHEN DMA REQUESTS ARE STILL
                                        ;PENDING.
                                        ;ROUTINE TEST
                                        ;.      IF UFD OR NO 2ND Q22BE THEN
                                        ;.      .       EXIT TEST
                                        ;.      ENDIF
                                        ;.      SET PMG COUNT TO SOME VALUE
                                        ;.      PROGRAM BOTH Q22BE TO INTERRUPT AT THE SAME LEVEL
                                        ;.      DETERMINE WHICH HAS HIGHER PRIORITY ON THE BUS
                                        ;.      PROGRAM Q22BE WITH HIGHER PRIORITY TO DO HOG MODE
                                        ;.      PROGRAM 2ND ONE TO DO A CYCLE
                                        ;.      CACHE INSTRUCTION THAT WILL STOP 2ND DMA
                                        ;.      INITIATE BOTH DMA CYCLES
                                        ;.      STOP 2ND DMA (RESET IS "STOLEN" CPU BUS CYCLE)
                                        ;.      IF 2ND DMA HAPPENED THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;.      CLEAR PMG COUNT
                                        ;.      PROGRAM Q22BE WITH HI ER PRIORITY TO DO HOG MODE
                                        ;.      PROGRAM 2ND ONE TO DO A CYCLE
                                        ;.      CACHE INSTRUCTION THAT WILL STOP 2ND DMA
                                        ;.      INITIATE BOTH DMA CYCLES
                                        ;.      STOP 2ND DMA (CLEARING OF CSR2<0> IS "STOLEN" CPU BUS CYCLE)
                                        ;.      IF 2ND DMA DIDN'T HAPPENED THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;ENDROUTINE

                                        ;;*****************************************************************
20462  121312  000004                   TST65:  SCOPE
20463  121314  032737  000040  000052           BIT     @#BIT05,@#52                    ;UFD MODE?
20464  121322  001402                           BEQ     100$                            ;IF NOT, CONTINUE
20465  121324  000137  132724                   JMP     $EOP                            ;EXIT
20466  121330  005737  002664           100$:   TST     CSR12                           ;SECOND Q22BE AVAILABLE?
20467  121334  001002                           BNE     110$                            ;IF YES, GO DO TEST
20468  121336  000137  132724                   JMP     $EOP                            ;OTHERWISE, GOTO EOP
20469                                   ;
20470                                   ; DETERMINE WHICH Q22BE IS CLOSER TO CPU BY DOING DATO
20471                                   ; THE CSR1 OF THE ONE WITH HIGHER PRIORITY IS IN R4, THE SECOND IN R2
20472                                   ;
20473  121342  005077  061300           110$:   CLR     @CSR2                           ;CLEAR JUST IN CASE
20474  121346  005077  061314                   CLR     @CSR22
20475  121352  012777  002710  061270           MOV     #TEMP,@BA                       ;ADDRESS TO BE USED
20476  121360  012777  002710  061302           MOV     #TEMP,@BA2                      ;FOR BOTH OF THEM
20477  121366  012777  177777  061256           MOV     #177777,@WC                     ;DO FOR 1 WORD
20478  121374  012777  177777  061270           MOV     #177777,@WC2                    ;IN BOTH Q22BW'S
20479  121402  012777  012525  061244           MOV     #12525,@DATA                    ;DATA FOR 1ST
20480  121410  012777  052525  061256           MOV     #52525,@DATA2                   ;DATA FOR 2ND
20481  121416  012777  001601  061220           MOV     #1601,@CSR1                     ;1 DATO FOR 1ST
20482  121424  012777  001601  061232           MOV     #1601,@CSR12                    ;AND 2ND
20483  121432  012777  000001  061222           MOV     #1,@SIMGOA                      ;BOTH GO
20484  121440  105777  061200           1$:     TSTB    @CSR1                           ;FIRST DONE?
20485  121444  100375                           BPL     1$                              ;IF NOT, WAIT
20486  121446  105777  061212           2$:     TSTB    @CSR12                          ;SECOND DONE
```

```
20487  121452  100375                          BPL     2$                      ;WAIT FOR 2ND
20488  121454  022737  052525  002710          CMP     #52525,TEMP             ;SECOND FINISHED LAST?
20489  121462  001405                          BEQ     3$                      ;IF SO, BRANCH
20490  121464  013704  002664                  MOV     CSR12,R4                ;SECOND ONE AT HIGHER LEVEL
20491  121470  013702  002644                  MOV     CSR1,R2                 ;FIRST AT LOWER
20492  121474  000404                          BR      4$                      ;DO PMG PART
20493  121476  013704  002644          3$:     MOV     CSR1,R4                 ;FIRST ONE AT HIGHER LEVEL
20494  121502  013702  002664                  MOV     CSR12,R2                ;SECOND AT LOWER
20495                                          ;
20496                                          ; INITIATE DMA CYCLES TO WORK WITH PMG COUNTER
20497                                          ;
20498  121506  013737  177520  002710  4$:     MOV     BCSR,SAVBR              ;STORE BCSR REGISTER
20499  121514  012700  000001                  MOV     #BIT00,R0               ;FIRST VALUE FOR PMG 0.4MSEC
20500  121520  050037  177520          5$:     BIS     R0,BCSR                 ;CHANGE PMG COMUTER
20501  121524  005737  121626                  TST     6$                      ;CACHE RESET INTRUCTION
20502  121530  005737  121630                  TST     6$+2                    ;AND THE FOLLOWING FEW
20503  121534  005737  121632                  TST     6$+4                    ;
20504  121540  005737  121634                  TST     6$+6                    ;
20505  121544  012714  001407                  MOV     #1407,(R4)              ;DATI IN HOG MODE FOR HIGHER ONE
20506  121550  005064  000002                  CLR     2(R4)                   ;CLEAR CSR2
20507  121554  012764  002710  000004          MOV     #TEMP,4(R4)             ;ADDRESS TO START DATI
20508  121562  012764  000000  000006          MOV     #0,6(R4)                ;DO 128 DATI'S IN HOG MODE
20509  121570  012712  001407                  MOV     #1407,(R2)              ;DATO FOR LOWER LEVEL ONE
20510  121574  005062  000002                  CLR     2(R2)                   ;CLEAR JUST IN CASE
20511  121600  012762  002710  000004          MOV     #TEMP,4(R2)             ;USE THE SAME ADDRESS
20512  121606  012762  177777  000006          MOV     #177777,6(R2)           ;DO JUST ONE DATO
20513  121614  005062  000010                  CLR     10(R2)                  ;CLEAR DATA OF 2ND Q22BE
20514  121620  012762  000001  000016          MOV     #1,16(R2)               ;BOTH GO
20515  121626  000005                  6$:     RESET                           ;STOP Q22BE AT R4
20516  121630  023762  002710  000010          CMP     TEMP,10(R2)             ;SECOND DMA DONE?
20517  121636  001001                          BNE     7$                      ;IF SET, BRANCH
20518  121640  104127                          ERROR   ^127                    ;NO CYCLE STEALING
20519  121642  062700  000003          7$:     ADD     #3,R0                   ;DO FOR 1,3,7 IN PMG
20520  121646  040037  177520                  BIC     R0,BCSR                 ;CLEAR PREVOIUS BITS IN BCSR
20521  121652  022700  000007                  CMP     #7,R0                   ;LAST ONE?
20522  121656  002320                          BGE     5$                      ;IF NOT, BRANCH
20523                                          ;
20524                                          ; TRY WITHOUT PMG COUNTER OPERATING
20525                                          ;
20526  121660  042737  000007  177520          BIC     #7,BCSR                 ;TURN OF PMG COUNTER
20527  121666  005737  121770                  TST     8$                      ;CACHE RESET FETCH
20528  121672  005737  121772                  TST     8$+2                    ;AND THE FOLLOWING FEW
20529  121676  005737  121774                  TST     8$+4                    ;
20530  121702  005737  121776                  TST     8$+6                    ;
20531  121706  012714  001407                  MOV     #1407,(R4)              ;DATI IN HOG MODE FOR HIGHER ONE
20532  121712  005064  000002                  CLR     2(R4)                   ;CLEAR CSR2
20533  121716  012764  002710  000004          MOV     #TEMP,4(R4)             ;ADDRESS TO START DATI
20534  121724  012764  000000  000006          MOV     #0,6(R4)                ;DO 128 DATI'S IN HOG MODE
20535  121732  012712  001407                  MOV     #1407,(R2)              ;DATO FOR LOWER LEVEL ONE
20536  121736  005062  000002                  CLR     2(R2)                   ;CLEAR CRS2 OF 2ND
20537  121742  012762  002710  000004          MOV     #TEMP,4(R2)             ;USE THE SAME ADDRESS
20538  121750  012762  177777  000006          MOV     #177777,6(R2)           ;DO JUST ONE DA1O
20539  121756  005062  000010                  CLR     10(R2)                  ;CLEAR DATA OF 2ND Q22BE
20540  121762  012777  000001  060672          MOV     #1,@SIMGOA              ;BOTH GO
20541  121770  000005                  8$:     RESET                           ;IF NOT WORKING, STOPS 2ND
20542  121772  023762  002710  000010          CMP     TEMP,10(R2)             ;2ND DMA HAPPENED?
```

```
 20543  122000  001401                          BEQ     9$                              ;IF YES, BRANCH
 20544  122002  104127                          ERROR   +127                            ;IN PMG COUNTER
 20545  122004  013737  002710  177520  9$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
 20546
 20547
 20548                                  ;;*************************************************************************
 20549  122012  000004                  TST66:  SCOPE
 20550
 20551  122014  000137  132724                  JMP     $EOP
```

```
 20552                                          .SBTTL  GLOBAL ERROR MESSAGES
 20553  122020  040502  044523  020103  EM1:    .ASCIZ  /BASIC INSTRUCTION SET ERROR/
 20554  122026  047111  052123  052522
 20555  122034  052103  047511  020116
 20556  122042  042523  020124  051105
 20557  122050  047522  000122
 20558  122054  046515  020125  051105  EM2:    .ASCIZ  /MMU ERROR/
 20559  122062  047522  000122
 20560  122066  050106  020120  051105  EM3:    .ASCIZ  /FPP ERROR/
 20561  122074  047522  000122
 20562  122100  051105  047522  020122  EM4:    .ASCIZ  /ERROR IN READ-WRITE BITS OF CCR/
 20563  122106  047111  051040  040505
 20564  122114  026504  051127  052111
 20565  122122  020105  044502  051524
 20566  122130  047440  020106  041503
 20567  122136  000122
 20568  122140  047506  041522  020105  EM5:    .ASCIZ  /FORCE MISS WRITES TO CACHE/
 20569  122146  044515  051523  053440
 20570  122154  044522  042524  020123
 20571  122162  047524  041440  041501
 20572  122170  042510     000
 20573  122173     106  051117  042503  EM6:    .ASCIZ  /FORCE MISS WRITE INVALIDATES CACHE/
 20574  122200  046440  051511  020123
 20575  122206  051127  052111  020105
 20576  122214  047111  040526  044514
 20577  122222  040504  042524  020123
 20578  122230  040503  044103  000105
 20579  122236  047125  054105  042520  EM7:    .ASCIZ  /UNEXPECTED PARITY INTERRUPT/
 20580  122244  052103  042105  050040
 20581  122252  051101  052111  020131
 20582  122260  047111  042524  051122
 20583  122266  050125  000124
 20584  122272  040524  020107  040520  EM10:   .ASCIZ  /TAG PARITY ERROR/
 20585  122300  044522  054524  042440
 20586  122306  051122  051117     000
 20587  122313     104  052101  020101  EM11:   .ASCIZ  /DATA PARITY ERROR/
 20588  122320  040520  044522  054524
 20589  122326  042440  051122  051117
 20590  122334     000
 20591  122335     114  053517  041040  EM12:   .ASCIZ  /LOW BYTE PARITY ERROR/
 20592  122342  052131  020105  040520
 20593  122350  044522  054524  042440
 20594  122356  051122  051117     000
 20595  122363     110  043511  020110  EM13:   .ASCIZ  /HIGH BYTE PARITY ERROR/
 20596  122370  054502  042524  050040
 20597  122376  051101  052111  020131
 20598  122404  051105  047522  000122
 20599  122412  051105  047522  020122  EM14:   .ASCIZ  /ERROR IN DATA PATH/
 20600  122420  047111  042040  052101
 20601  122426  020101  040520  044124
 20602  122434     000
 20603  122435     106  051117  042503  EM15:   .ASCIZ  /FORCE MISS READS FROM CACHE/
 20604  122442  046440  051511  020123
 20605  122450  042522  042101  020123
 20606  122456  051106  046517  041440
 20607  122464  041501  042510     000
```

```
 20608  122471     106  051117  042503  EM16:   .ASCIZ  /FORCE MISS READS FROM CACHE AND MISS/
 20609  122476  046440  051511  020123
 20610  122504  042522  042101  020123
 20611  122512  051106  046517  041440
 20612  122520  041501  042510  040440
 20613  122526  042116  046440  051511
 20614  122534  000123
 20615  122536  051105  047522  020122  EM17:   .ASCIZ  \ERROR IN RECORDING HITS IN HIT/MISS\
 20616  122544  047111  051040  041505
 20617  122552  051117  044504  043516
 20618  122560  044040  052111  020123
 20619  122566  047111  044040  052111
 20620  122574  046457  051511  000123
 20621  122602  051127  052111  020105  EM20:   .ASCIZ  /WRITE BYTE ALLOCATES CACHE/
 20622  122610  054502  042524  040440
 20623  122616  046114  041517  052101
 20624  122624  051505  041440  041501
 20625  122632  042510     000
 20626  122635     127  044522  042524  EM21:   .ASCIZ  /WRITE BYTE HIT DOES NOT RECORD HIT/
 20627  122642  041040  052131  020105
 20628  122650  044510  020124  047504
 20629  122656  051505  047040  052117
 20630  122664  051040  041505  051117
 20631  122672  020104  044510  000124
 20632  122700  054502  042524  020123  EM22:   .ASCIZ  /BYTES REVERSED ON WRITE CYCLES/
 20633  122706  042522  042526  051522
 20634  122714  042105  047440  020116
 20635  122722  051127  052111  020105
 20636  122730  054503  046103  051505
 20637  122736     000
 20638  122737     103  047117  044504  EM23:   .ASCIZ  /CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE/
 20639  122744  044524  047117  046101
 20640  122752  041040  050131  051501
 20641  122760  020123  047504  051505
 20642  122766  023516  020124  047111
 20643  122774  040526  044514  040504
 20644  123002  042524  041440  041501
 20645  123010  042510     000
 20646  123013     110  052111  020123  EM24:   .ASCIZ  /HITS RECORDED AFTER FLUSHING CACHE/
 20647  123020  042522  047503  042122
 20648  123026  042105  040440  052106
 20649  123034  051105  043040  052514
 20650  123042  044123  047111  020107
 20651  123050  040503  044103  000105
 20652  123056  054502  040520  051523  EM25:   .ASCIZ  /BYPASS DOESN'T INVALIDATE CACHE/
 20653  123064  042040  042517  047123
 20654  123072  052047  044440  053116
 20655  123100  046101  042111  052101
 20656  123106  020105  040503  044103
 20657  123114  000105
 20658  123116  051515  051105  042040  EM26:   .ASCIZ  /MSER DOES NOT CLEAR ON WRITE REFERENCE/
 20659  123124  042517  020123  047516
 20660  123132  020124  046103  040505
 20661  123140  020122  047117  053440
 20662  123146  044522  042524  051040
 20663  123154  043105  051105  047105
```

```
20664   123162  042503     000
20665   123165     120  051101  052111  EM27:   .ASCIZ  /PARITY ERROR DON'T CAUSE A MISS/
20666   123172  020131  051105  047522
20667   123200  020122  047504  023516
20668   123206  020124  040503  051525
20669   123214  020105  020101  044515
20670   123222  051523     000
20671   123225     120  051101  052111  EM30:   .ASCIZ  /PARITY ERROR DON'T SET MSER WITH CCR<7>=0/
20672   123232  020131  051105  047522
20673   123240  020122  047504  023516
20674   123246  020124  042523  020124
20675   123254  051515  051105  053440
20676   123262  052111  020110  041503
20677   123270  036122  037067  030075
20678   123276     000
20679   123277     120  051101  052111  EM31:   .ASCIZ  /PARITY ERROR IGNORED/
20680   123304  020131  051105  047522
20681   123312  020122  043511  047516
20682   123320  042522  000104
20683   123324  040520  044522  054524  EM32:   .ASCIZ  /PARITY ERROR IGNORED ON LOW BYTE/
20684   123332  042440  051122  051117
20685   123340  044440  047107  051117
20686   123346  042105  047440  020116
20687   123354  047514  020127  054502
20688   123362  042524     000
20689   123365     120  051101  052111  EM33:   .ASCIZ  /PARITY ERROR IGNORED ON HIGH BYTE/
20690   123372  020131  051105  047522
20691   123400  020122  043511  047516
20692   123406  042522  020104  047117
20693   123414  044040  043511  020110
20694   123422  054502  042524     000
20695   123427     120  051101  052111  EM34:   .ASCIZ  /PARITY ABORT LOGIC DOESN'T WORK/
20696   123434  020131  041101  051117
20697   123442  020124  047514  044507
20698   123450  020103  047504  051505
20699   123456  023516  020124  047527
20700   123464  045522     000
20701   123467     115  042523  020122  EM35:   .ASCIZ  /MSER NOT SET PROPERLY/
20702   123474  047516  020124  042523
20703   123502  020124  051120  050117
20704   123510  051105  054514     000
20705   123515     120  051101  052111  EM36:   .ASCIZ  /PARITY INTERRUPT LOGIC DOESN'T WORK/
20706   123522  020131  047111  042524
20707   123530  051122  050125  020124
20708   123536  047514  044507  020103
20709   123544  047504  051505  023516
20710   123552  020124  047527  045522
20711   123560     000
20712   123561     116  046530  040440  EM37:   .ASCIZ  /NXM AND PARITY ABORT DIN'T HAPPEN/
20713   123566  042116  050040  051101
20714   123574  052111  020131  041101
20715   123602  051117  020124  044504
20716   123610  023516  020124  040510
20717   123616  050120  047105     000
20718   123623     120  051101  052111  EM40:   .ASCIZ  /PARITY ABORT NOT BLOCKED BY NXM TRAP/
20719   123630  020131  041101  051117
```

```
 20720  123636  020124  047516  020124
 20721  123644  046102  041517  042513
 20722  123652  020104  054502  047040
 20723  123660  046530  052040  040522
 20724  123666  000120
 20725  123670  052515  052114  026511  EM41:   .ASCIZ  /MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS/
 20726  123676  051120  041517  051505
 20727  123704  047523  020122  047510
 20728  123712  045517  044440  051516
 20729  123720  051124  041525  044524
 20730  123726  047117  042040  042517
 20731  123734  047123  052047  041440
 20732  123742  052501  042523  046440
 20733  123750  051511  000123
 20734  123754  051105  047522  020122  EM42:   .ASCIZ  /ERROR IN PARITY LOGIC/
 20735  123762  047111  050040  051101
 20736  123770  052111  020131  047514
 20737  123776  044507  000103
 20738  124002  051105  047522  020122  EM43:   .ASCIZ  /ERROR IN CACHE DATA RAMS/
 20739  124010  047111  041440  041501
 20740  124016  042510  042040  052101
 20741  124024  020101  040522  051515
 20742  124032     000
 20743  124033     105  051122  051117  EM44:   .ASCIZ  /ERROR IN NXM IN STANDALONE MODE/
 20744  124040  044440  020116  054116
 20745  124046  020115  047111  051440
 20746  124054  040524  042116  046101
 20747  124062  047117  020105  047515
 20748  124070  042504     000
 20749  124073     105  051122  051117  EM45:   .ASCIZ  \ERROR IN RECORDING HITS THROUGH HIT/MISS REGISTER\
 20750  124100  044440  020116  042522
 20751  124106  047503  042122  047111
 20752  124114  020107  044510  051524
 20753  124122  052040  051110  052517
 20754  124130  044107  044040  052111
 20755  124136  046457  051511  020123
 20756  124144  042522  044507  052123
 20757  124152  051105     000
 20758  124155     110  052111  051040  EM46:   .ASCIZ  /HIT RECORDED FOR A LOCATION THAT SHOULD NOT BE IN CACHE/
 20759  124162  041505  051117  042504
 20760  124170  020104  047506  020122
 20761  124176  020101  047514  040503
 20762  124204  044524  047117  052040
 20763  124212  040510  020124  044123
 20764  124220  052517  042114  047040
 20765  124226  052117  041040  020105
 20766  124234  047111  041440  041501
 20767  124242  042510     000
 20768  124245     115  051511  020123  EM47:   .ASCIZ  /MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE/
 20769  124252  042522  047503  042122
 20770  124260  042105  043040  051117
 20771  124266  040440  046040  041517
 20772  124274  052101  047511  020116
 20773  124302  044124  052101  051440
 20774  124310  047510  046125  020104
 20775  124316  042502  044440  020116
```

```
 20776  124324  040503  044103  000105
 20777  124332  051105  047522  020122  EM50:   .ASCIZ  /ERROR IN TAG STORE/
 20778  124340  047111  052040  043501
 20779  124346  051440  047524  042522
 20780  124354     000
 20781  124355     105  051122  051117  EM51:   .ASCIZ  /ERROR PCR READ-WRITE BITS/
 20782  124362  050040  051103  051040
 20783  124370  040505  026504  051127
 20784  124376  052111  020105  044502
 20785  124404  051524     000
 20786  124407     105  051122  051117  EM52:   .ASCIZ  /ERROR IN BCSR READ-WRITE BITS/
 20787  124414  044440  020116  041502
 20788  124422  051123  051040  040505
 20789  124430  026504  051127  052111
 20790  124436  020105  044502  051524
 20791  124444     000
 20792  124445     122  051505  052105  EM53:   .ASCIZ  /RESET DOESN'T CLEAR BCSR<4>/
 20793  124452  042040  042517  047123
 20794  124460  052047  041440  042514
 20795  124466  051101  041040  051503
 20796  124474  036122  037064     000
 20797  124501     103  042510  045503  EM54:   .ASCIZ  /CHECKSUM ERROR IN 16-BIT ROM /
 20798  124506  052523  020115  051105
 20799  124514  047522  020122  047111
 20800  124522  030440  026466  044502
 20801  124530  020124  047522  020115
 20802  124536     000
 20803  124537     103  042510  045503  EM55:   .ASCIZ  /CHECKSUM ERROR IN 8-BIT ROM/
 20804  124544  052523  020115  051105
 20805  124552  047522  020122  047111
 20806  124560  034040  041055  052111
 20807  124566  051040  046517     000
 20808  124573     124  046511  047505  EM56:   .ASCIZ  /TIMEOUT READING LKS/
 20809  124600  052125  051040  040505
 20810  124606  044504  043516  046040
 20811  124614  051513     000
 20812  124617     114  051513  030074  EM57:   .ASCIZ  /LKS<07> DOES NOT BECOME 1/
 20813  124624  037067  042040  042517
 20814  124632  020123  047516  020124
 20815  124640  042502  047503  042515
 20816  124646  030440     000
 20817  124651     127  044522  042524  EM60:   .ASCIZ  /WRITE REFERENCE DOESN'T CLEAR LKS<07>/
 20818  124656  051040  043105  051105
 20819  124664  047105  042503  042040
 20820  124672  042517  047123  052047
 20821  124700  041440  042514  051101
 20822  124706  046040  051513  030074
 20823  124714  037067     000
 20824  124717     111  046114  043505  EM61:   .ASCIZ  /ILLEGAL LKS INTERRUPTS/
 20825  124724  046101  046040  051513
 20826  124732  044440  052116  051105
 20827  124740  052522  052120  000123
 20828  124746  051120  041517  051505  EM62:   .ASCIZ  /PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>/
 20829  124754  047523  020122  047111
 20830  124762  042524  051122  050125
 20831  124770  051524  042040  047117
```

```
 20832  124776  052047  041440  042514
 20833  125004  051101  046040  051513
 20834  125012  030074  037067     000
 20835  125017     114  051513  051040  EM63:   .ASCIZ  /LKS READY DOESN'T GO LOW/
 20836  125024  040505  054504  042040
 20837  125032  042517  047123  052047
 20838  125040  043440  020117  047514
 20839  125046  000127
 20840  125050  051127  047117  020107  EM64:   .ASCIZ  /WRONG NUMBER OF LKS INTERRUPTS/
 20841  125056  052516  041115  051105
 20842  125064  047440  020106  045514
 20843  125072  020123  047111  042524
 20844  125100  051122  050125  051524
 20845  125106     000
 20846  125107     114  051513  044440  EM65:   .ASCIZ  /LKS INTERRUPTS HAPPEN AT WRONG PRIORITY/
 20847  125114  052116  051105  052522
 20848  125122  052120  020123  040510
 20849  125130  050120  047105  040440
 20850  125136  020124  051127  047117
 20851  125144  020107  051120  047511
 20852  125152  044522  054524     000
 20853  125157     102  051503  036122  EM66:   .ASCIZ  /BCSR<12> DOES NOT DISABLES LKS/
 20854  125164  031061  020076  047504
 20855  125172  051505  047040  052117
 20856  125200  042040  051511  041101
 20857  125206  042514  020123  045514
 20858  125214  000123
 20859  125216  041502  051123  030474  EM67:   .ASCIZ  /BCSR<13> DOESN'T SET LKS<06>/
 20860  125224  037063  042040  042517
 20861  125232  047123  052047  051440
 20862  125240  052105  046040  051513
 20863  125246  030074  037066     000
 20864  125253     122  051505  052105  EM70:   .ASCIZ  /RESET DOESN'T SET LKS<7>/
 20865  125260  042040  042517  047123
 20866  125266  052047  051440  052105
 20867  125274  046040  051513  033474
 20868  125302  000076
 20869  125304  042522  042523  020124  EM71:   .ASCIZ  /RESET DOESN'T CLEAR LKS<06>/
 20870  125312  047504  051505  023516
 20871  125320  020124  046103  040505
 20872  125326  020122  045514  036123
 20873  125334  033060  000076
 20874  125340  044524  042515  052517  EM72:   .ASCIZ  /TIMEOUT READING SLU REGISTERS/
 20875  125346  020124  042522  042101
 20876  125354  047111  020107  046123
 20877  125362  020125  042522  044507
 20878  125370  052123  051105  000123
 20879  125376  051105  047522  020122  EM73:   .ASCIZ  /ERROR IN XMIT READY/
 20880  125404  047111  054040  044515
 20881  125412  020124  042522  042101
 20882  125420  000131
 20883  125422  041522  051123  033474  EM74:   .ASCIZ  /RCSR<7> DOESN'T BECOME 1/
 20884  125430  020076  047504  051505
 20885  125436  023516  020124  042502
 20886  125444  047503  042515  030440
 20887  125452     000
```

```
 20888  125453     127  047522  043516  EM75:    .ASCIZ  /WRONG CHARACTER RECEIVED/
 20889  125460  041440  040510  040522
 20890  125466  052103  051105  051040
 20891  125474  041505  044505  042526
 20892  125502  000104
 20893  125504  041522  051123  030074  EM76:    .ASCIZ  /RCSR<07> NOT CLEARED AFTER READING RBUF/
 20894  125512  037067  047040  052117
 20895  125520  041440  042514  051101
 20896  125526  042105  040440  052106
 20897  125534  051105  051040  040505
 20898  125542  044504  043516  051040
 20899  125550  052502  000106
 20900  125554  041530  051123  030074  EM77:    .ASCIZ  /XCSR<07> NOT SET ON RESET/
 20901  125562  037067  047040  052117
 20902  125570  051440  052105  047440
 20903  125576  020116  042522  042523
 20904  125604  000124
 20905  125606  041522  051123  030074  EM100:   .ASCIZ  /RCSR<07> NOT CLEARED ON RESET/
 20906  125614  037067  047040  052117
 20907  125622  041440  042514  051101
 20908  125630  042105  047440  020116
 20909  125636  042522  042523  000124
 20910  125644  046123  020125  047111  EM101:   .ASCIZ  /SLU INTERRUPTS HAPPEN AT 4/
 20911  125652  042524  051122  050125
 20912  125660  051524  044040  050101
 20913  125666  042520  020116  052101
 20914  125674  032040     000
 20915  125677     122  051505  052105  EM102:   .ASCIZ  /RESET DOES NOT CLEAR PROPER BITS IN SLU REGISTERS/
 20916  125704  042040  042517  020123
 20917  125712  047516  020124  046103
 20918  125720  040505  020122  051120
 20919  125726  050117  051105  041040
 20920  125734  052111  020123  047111
 20921  125742  051440  052514  051040
 20922  125750  043505  051511  042524
 20923  125756  051522     000
 20924  125761     124  040522  051516  EM103:   .ASCIZ  /TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>/
 20925  125766  044515  020124  047111
 20926  125774  042524  051122  050125
 20927  126002  020124  047504  051505
 20928  126010  047040  052117  041440
 20929  126016  042514  051101  054040
 20930  126024  051503  036122  033460
 20931  126032  000076
 20932  126034  042522  042503  053111  EM104:   .ASCIZ  /RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>/
 20933  126042  020105  047111  042524
 20934  126050  051122  050125  051524
 20935  126056  042040  047117  052047
 20936  126064  041440  042514  051101
 20937  126072  051040  051503  036122
 20938  126100  033460  000076
 20939  126104  051102  040505  020113  EM105:   .ASCIZ  /BREAK CONDITION DOES NOT SET RBUF PROPERLY/
 20940  126112  047503  042116  052111
 20941  126120  047511  020116  047504
 20942  126126  051505  047040  052117
 20943  126134  051440  052105  051040
```

```
20944   126142  052502  020106  051120
20945   126150  050117  051105  054514
20946   126156     000
20947   126157     122  052502  020106  EM106:  .ASCIZ  /RBUF <15-11> WASN'T CLEARED ON NEXT CHARACTER/
20948   126164  030474  026465  030461
20949   126172  020076  040527  047123
20950   126200  052047  041440  042514
20951   126206  051101  042105  047440
20952   126214  020116  042516  052130
20953   126222  041440  040510  040522
20954   126230  052103  051105     000
20955   126235     123  052514  044440  EM107:  .ASCIZ  /SLU INTERRUPTS DON'T HAPPNEN/
20956   126242  052116  051105  052522
20957   126250  052120  020123  047504
20958   126256  023516  020124  040510
20959   126264  050120  042516  000116
20960   126272  051105  047522  020122  EM110:  .ASCIZ  /ERROR IN WRITING TO SLU REGISTERS/
20961   126300  047111  053440  044522
20962   126306  044524  043516  052040
20963   126314  020117  046123  020125
20964   126322  042522  044507  052123
20965   126330  051105  000123
20966   126334  044506  051522  020124  EM111:  .ASCIZ  /FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND/
20967   126342  044103  051101  041501
20968   126350  042524  020122  040527
20969   126356  020123  047516  020124
20970   126364  053117  051105  052522
20971   126372  020116  054502  052040
20972   126400  042510  051440  041505
20973   126406  047117  000104
20974   126412  053117  051105  052522  EM112:  .ASCIZ  /OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF/
20975   126420  020116  047503  042116
20976   126426  052111  047511  020116
20977   126434  047504  051505  047040
20978   126442  052117  051440  052105
20979   126450  050040  047522  042520
20980   126456  020122  044502  051524
20981   126464  044440  020116  041122
20982   126472  043125     000
20983   126475     117  042526  051122  EM113:  .ASCIZ  /OVERRUN BITS WERE NOT CLEARED ON THE NEXT CHARACTER/
20984   126502  047125  041040  052111
20985   126510  020123  042527  042522
20986   126516  047040  052117  041440
20987   126524  042514  051101  042105
20988   126532  047440  020116  044124
20989   126540  020105  042516  052130
20990   126546  041440  040510  040522
20991   126554  052103  051105     000
20992   126561     105  051122  051117  EM114:  .ASCIZ  \ERROR ON XCSR<2>\
20993   126566  047440  020116  041530
20994   126574  051123  031074  000076
20995   126602  051105  047522  020122  EM115:  .ASCIZ  /ERROR IN TAG STORE FROM STANDALONE MODE/
20996   126610  047111  052040  043501
20997   126616  051440  047524  042522
20998   126624  043040  047522  020115
20999   126632  052123  047101  040504
```

```
21000   126640  047514  042516  046440
21001   126646  042117  000105
21002   126652  046504  020101  040524  EM116:  .ASCIZ  /DMA TAG PARITY DOES NOT GENERATE PROPER RESPONSE/
21003   126660  020107  040520  044522
21004   126666  054524  042040  042517
21005   126674  020123  047516  020124
21006   126702  042507  042516  040522
21007   126710  042524  050040  047522
21008   126716  042520  020122  042522
21009   126724  050123  047117  042523
21010   126732     000
21011   126733     115  042523  036122  EM117:  .ASCIZ  /MSER<13>NOT SET IN STANDALONE MODE/
21012   126740  031461  047076  052117
21013   126746  051440  052105  044440
21014   126754  020116  052123  047101
21015   126762  040504  047514  042516
21016   126770  046440  042117  000105
21017   126776  046504  020101  051127  EM120:  .ASCIZ  /DMA WRITE HITS DON'T INVALIDATE CACHE/
21018   127004  052111  020105  044510
21019   127012  051524  042040  047117
21020   127020  052047  044440  053116
21021   127026  046101  042111  052101
21022   127034  020105  040503  044103
21023   127042  000105
21024   127044  047111  041040  047514  EM121:  .ASCIZ  /IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED/
21025   127052  045503  046440  042117
21026   127060  020105  047117  053440
21027   127066  044522  042524  042040
21028   127074  040515  044040  052111
21029   127102  020123  047516  020124
21030   127110  053105  051105  052131
21031   127116  044510  043516  044440
21032   127124  020123  047111  040526
21033   127132  044514  040504  042524
21034   127140  000104
21035   127142  042522  042101  042040  EM122:  .ASCIZ  /READ DMA HIT IS WRONG/
21036   127150  040515  044040  052111
21037   127156  044440  020123  051127
21038   127164  047117  000107
21039   127170  051105  047522  020122  EM123:  .ASCIZ  /ERROR IN Q22BE DMA CYCLES/
21040   127176  047111  040440  031062
21041   127204  042502  042040  040515
21042   127212  041440  041531  042514
21043   127220  000123
21044   127222  044520  050522  044440  EM124:  .ASCIZ  /PIRQ INTERRUOTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS/
21045   127230  052116  051105  052522
21046   127236  052117  020123  047504
21047   127244  023516  020124  040524
21048   127252  042513  050040  044522
21049   127260  051117  052111  020131
21050   127266  053117  051105  050440
21051   127274  041040  051525  044440
21052   127302  052116  051105  052522
21053   127310  052120  000123
21054   127314  047516  050040  053517  EM125:  .ASCIZ  /NO POWER DOWN TRAP TO 24 OCCUR/
21055   127322  051105  042040  053517
```

```
 21056  127330  020116  051124  050101
 21057  127336  052040  020117  032062
 21058  127344  047440  041503  051125
 21059  127352     000
 21060  127353     105  051122  051117  EM126:  .ASCIZ  /ERROR DOING Q22BE INTERRUPTS/
 21061  127360  042040  044517  043516
 21062  127366  050440  031062  042502
 21063  127374  044440  052116  051105
 21064  127402  052522  052120  000123
 21065  127410  051105  047522  020122  EM127:  .ASCIZ  /ERROR IN OPERATION OF PMG COUNTER/
 21066  127416  047111  047440  042520
 21067  127424  040522  044524  047117
 21068  127432  047440  020106  046520
 21069  127440  020107  047503  047125
 21070  127446  042524  000122
 21071  127452  047125  054105  042520  EM130:  .ASCIZ  /UNEXPECTED TRAP TO 4/
 21072  127460  052103  042105  052040
 21073  127466  040522  020120  047524
 21074  127474  032040     000
 21075  127477     105  051122  051117  EM131:  .ASCIZ  /ERROR WRITING TO LKS<6>/
 21076  127504  053440  044522  044524
 21077  127512  043516  052040  020117
 21078  127520  045514  036123  037066
 21079  127526     000
 21080  127527     105  051122  051117  EM132:  .ASCIZ  /ERROR IN MAITENANCE REGISTER/
 21081  127534  044440  020116  040515
 21082  127542  052111  047105  047101
 21083  127550  042503  051040  043505
 21084  127556  051511  042524  000122
 21085
 21086  127564  052040  051505  004524  DH1:    .ASCII  / TEST  ERROR/<15><12>
 21087  127572  051105  047522  006522
 21088  127600     012
 21089  127601     040  021440  020011          .ASCIZ  /  #     PC/
 21090  127606  041520     000
 21091  127611     040  042524  052123  DH4:    .ASCII  / TEST  ERROR   EXPCTED RECEIVED/<15><12>
 21092  127616  042411  051122  051117
 21093  127624  042411  050130  052103
 21094  127632  042105  051011  041505
 21095  127640  044505  042526  006504
 21096  127646     012
 21097  127647     040  021440  020011          .ASCIZ  /  #     PC      DATA     )ATA/
 21098  127654  041520  020011  040504
 21099  127662  040524  020040  020040
 21100  127670  042040  052101  000101
 21101  127676  052040  051505  004524  DH5:    .ASCII  / TEST  ERROR   HITMIS  DATA IN DATA IN/<15><12>
 21102  127704  051105  047522  004522
 21103  127712  044510  046524  051511
 21104  127720  042011  052101  020101
 21105  127726  047111  042011  052101
 21106  127734  020101  047111  005015
 21107  127742  020040  004443  050040          .ASCIZ  /  #     PC      REG.    CACHE   MEMORY/
 21108  127750  004503  051040  043505
 21109  127756  004456  040503  044103
 21110  127764  004505  042515  047515
 21111  127772  054522     000
```

```
21112  127775     040  042524  052123  DH7:    .ASCII  / TEST  ERROR   ADDRESS MSER/<15><12>
21113  130002  042411  051122  051117
21114  130010  040411  042104  042522
21115  130016  051523  046411  042523
21116  130024  006522     012
21117  130027     040  021440  020011          .ASCIZ  /  #     PC      ACCESSED/
21118  130034  041520  040411  041503
21119  130042  051505  042523  000104
21120  130050  052040  051505  004524  DH24:   .ASCII  / TEST  ERROR   NUMBER/<15><12>
21121  130056  051105  047522  004522
21122  130064  052516  041115  051105
21123  130072  005015
21124  130074  020040  004443  050040          .ASCIZ  /  #     PC      OF HITS/
21125  130102  004503  043117  044040
21126  130110  052111  000123
21127  130114  051105  047522  004522  DH27:   .ASCII  \ERROR  ERROR   MSER    HIT/MISS\<15><12>
21128  130122  051105  047522  004522
21129  130130  051515  051105  044011
21130  130136  052111  046457  051511
21131  130144  006523     012
21132  130147     040  021440  020011          .ASCIZ  /  #     PC/
21133  130154  041520     000
21134  130157     040  042524  052123  DH41:   .ASCII  / TEST  ERROR   INSTRUCTION/<15><12>
21135  130164  042411  051122  051117
21136  130172  044411  051516  051124
2113?  130200  041525  044524  047117
211?   130206  005015
21139  130210  020040  004443  050040          .ASCIZ  /  #     PC       OPCODE/
21140  130216  004503  047440  041520
21141  130224  042117  000105
21142  130230  052040  051505  004524  DH43:   .ASCII  / TEST  ERROR   EXPECTD RECEIVD CACHE/<15><12>
21143  130236  051105  047522  004522
21144  130244  054105  042520  052103
21145  130252  004504  042522  042503
21146  130260  053111  004504  040503
21147  130266  044103  006505     012
21148  130273     040  021440  020011          .ASCIZ  /  #     PC      DATA    DATA    LOCATION/
21149  130300  041520  020011  040504
21150  130306  040524  020011  040504
21151  130314  040524  046011  041517
21152  130322  052101  047511  000116
21153  130330  052040  051505  004524  DH47:   .ASCII  / TEST  ERROR   ADDRESS ADDRESS/<15><12>
21154  130336  051105  047522  004522
21155  130344  042101  051104  051505
21156  130352  004523  042101  051104
21157  130360  051505  006523     012
21158  130365     040  021440  020011          .ASCIZ  /  #     PC      <21-16>  <15-0>/
21159  130372  041520  036011  030462
21160  130400  030455  037066  020011
21161  130406  030474  026465  037060
21162  130414     000
21163  130415     040  042524  052123  DH65:   .ASCII  / TEST  ERROR   PRIORITY/<15><12>
21164  130422  042411  051122  051117
21165  130430  050011  044522  051117
21166  130436  052111  006531     012
21167  130443     040  021440  020011          .ASCIZ  /  #     PC      LEVEL/
```

```
 21168  130450  041520  046011  053105
 21169  130456  046105     000
 21170  130461     040  042524  052123  DH72:   .ASCII  / TEST  ERROR   ADDRESS/<15><12>
 21171  130466  042411  051122  051117
 21172  130474  040411  042104  042522
 21173  130502  051523  005015
 21174  130506  020040  004443  050040          .ASCIZ  /  #     PC     FAILED/
 21175  130514  004503  040506  046111
 21176  130522  042105     000
 21177  130525     040  042524  052123  DH105:  .ASCII  / TEST  ERROR   RBUF/<15><12>
 21178  130532  042411  051122  051117
 21179  130540  051011  052502  006506
 21180  130546     012
 21181  130547     040  021440  020011          .ASCIZ  /  #     PC/
 21182  130554  041520     000
 21183  130557     040  042524  052123  DH115:  .ASCII  / TEST  ERROR   MSER    ADDRESS/<15><12>
 21184  130564  042411  051122  051117
 21185  130572  046411  042523  004522
 21186  130600  042101  051104  051505
 21187  130606  006523     012
 21188  130611     040  021440  020011          .ASCIZ  /  #     PC             ACCESSED/
 21189  130616  041520  004440  040411
 21190  130624  041503  051505  042523
 21191  130632  000104
 21192                                          .EVEN
 21193  130634  001162  001116  000000  DT1:    .WORD   $TMP1,$ERRPC,0
 21194  130642  001162  001116  000001  DT4:    .WORD   $TMP1,$ERRPC,R1,CCR,0
 21195  130650  177746  000000
 21196  130654  001162  001116  000002  DT5:    .WORD   $TMP1,$ERRPC,R2,R1,$GDDAT,0
 21197  130662  000001  001124  000000
 21198  130670  001162  001116  001122  DT7:    .WORD   $TMP1,$ERRPC,$BDADR,MSER,0
 21199  130676  177744  000000
 21200  130702  001162  001116  001124  DT14:   .WORD   $TMP1,$ERRPC,$GDDAT,TSTLOC,0
 21201  130710  003122  000000
 21202  130714  001162  001116  001124  DT17:   .WORD   $TMP1,$ERRPC,$GDDAT,RECDAT,0
 21203  130722  110152  000000
 21204  130726  001162  001116  000003  DT24:   .WORD   $TMP1,$ERRPC,R3,0
 21205  130734  000000
 21206  130736  001162  001116  177744  DT27:   .WORD   $TMP1,$ERRPC,MSER,R3,0
 21207  130744  000003  000000
 21208  130750  001162  001116  001124  DT35:   .WORD   $TMP1,$ERRPC,$GDDAT,MSER,0
 21209  130756  177744  000000
 21210  130762  001162  001116  001126  DT41:   .WORD   $TMP1,$ERRPC,$BDDAT,0
 21211  130770  000000
 21212  130772  001162  001116  000001  DT43:   .WORD   $TMP1,$ERRPC,R1,RECDAT,$BDADR,0
 21213  131000  110152  001122  000000
 21214  131006  001162  001116  172354  DT47:   .WORD   $TMP1,$ERRPC,KIPAR6,$BDADR,0
 21215  131014  001122  000000
 21216  131020  001162  001116  000001  DT50:   .WORD   $TMP1,$ERRPC,R1,$BDADR,0
 21217  131026  001122  000000
 21218  131032  001162  001116  001124  DT51:   .WORD   $TMP1,$ERRPC,$GDDAT,PCR,0
 21219  131040  177522  000000
 21220  131044  001162  001116  001124  DT52:   .WORD   $TMP1,$ERRPC,$GDDAT,BCSR,0
 21221  131052  177520  000000
 21222  131056  001162  001116  001124  DT64:   .WORD   $TMP1,$ERRPC,$GDDAT,LKSFL,0
 21223  131064  002702  000000
```

```
 21224  131070  001162  001116  001124  DT65:   .WORD   $TMP1,$ERRPC,$GDDAT,0
 21225  131076  000000
 21226  131100  001162  001116  001124  DT75:   .WORD   $TMP1,$ERRPC,$GDDAT,$BDDAT,0
 21227  131106  001126  000000
 21228  131112  001162  001116  177562  DT105:  .WORD   $TMP1,$ERRPC,RBUF,0
 21229  131120  000000
 21230  131122  001162  001116  001126  DT115:  .WORD   $TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR,0
 21231  131130  172354  001122  000000
 21232  131136  001162  001122  000000  DT130:  .WORD   $TMP1,$BDADR,0
```

```
 21233                                  .SBTTL  MODIFIED ERROR MESSAGE TYPEOUT ROUTINE
 21234                                  ;;**********************************************************************
 21235                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 21236                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 21237                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 21238                                  ;*
 21239                                  ;*THE ONLY DIFFERENCE BETWEEN THIS ROUTINE AND THE ORIGINAL "$ERRTYP" FROM
 21240                                  ;*SYSMAC IS THAT YOU CAN PASS INFORMATION IN GENERAL PURPOSE REGISTERS TO THIS
 21241                                  ;*ROUTINE. THE GENERAL PURPOSE REGISTERS USED ARE TO BE SPECIFIED IN DT*
 21242                                  ;*FORMAT. R0 SHOULD NOT BE USED.
 21243
 21244  131144                          ERTYPE:
 21245  131144  005037  001162                 CLR     $TMP1           ;;JUST CLEAR IT
 21246  131150  113737  001102  001162         MOVB    $TSTNM,$TMP1    ;;STORE TEST NUMBER
 21247  131156  104401  001175                 TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 21248  131162  010046                         MOV     R0,-(SP)        ;;SAVE R0
 21249  131164  005000                         CLR     R0              ;;PICKUP THE ITEM INDEX
 21250  131166  153700  001114                 BISB    @#$ITEMB,R0
 21251  131172  001004                         BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 21252                                                                  ;;TYPE THE PC OF THE ERROR
 21253  131174  013746  001116                 MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 21254                                                                  ;;ERROR ADDRESS
 21255  131200  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 21256  131202  000426                         BR      6$              ;;GET OUT
 21257  131204  005300                 1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 21258  131206  006300                         ASL     R0              ;;      WORK FOR THE ERROR TABLE
 21259  131210  006300                         ASL     R0
 21260  131212  006300                         ASL     R0
 21261  131214  062700  001324                 ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 21262  131220  012037  131230                 MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 21263  131224  001404                         BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 21264  131226  104401                         TYPE
 21265  131230  000000                 2$:     .WORD   0               ;;ERROR MESSAGE POINTER GOES HERE
 21266  131232  104401  001175                 TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 21267  131236  012037  131246         3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
 21268  131242  001404                         BEQ     5$              ;;SKIP TYPEOUT IF 0
 21269  131244  104401                         TYPE                    ;;TYPE THE "DATA HEADER"
 21270  131246  000000                 4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
 21271  131250  104401  001175                 TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 21272  131254  011000                 5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
 21273  131256  001004                         BNE     7$              ;;GO TYPE THE DATA
 21274  131260  012600                 6$:     MOV     (SP)+,R0        ;;RESTORE R0
 21275  131262  104401  001175                 TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 21276  131266  000207                         RTS     PC              ;;RETURN
 21277  131270                         7$:
 21278  131270  021027  000005                 CMP     (R0),#5         ;;GENERAL PURPOSE REGISTER?
 21279  131274  101021                         BHI     9$              ;;IF NOT, GO TYPE DATA
 21280  131276  042737  000700  131332         BIC     #BIT8!BIT7!BIT6,8$      ;;CLEAR BITS FOR SOURCE REGISTER
 21281  131304  011037  001160                 MOV     (R0),$TMP0      ;;SAVE (R0)
 21282  131310  000337  001160                 SWAB    $TMP0           ;;GET REGISTER NUMBER TO HIGH BYTE
 21283  131314  006237  001160                 ASR     $TMP0           ;;GET REGISTER NUMBER TO BITS 8-6
 21284  131320  006237  001160                 ASR     $TMP0
 21285  131324  053737  001160  131332         BIS     $TMP0,8$        ;;SET BITS IN MOV INSTRUCTION
 21286                                                                  ;;ACCORDING TO REGISTER NUMBER
 21287  131332  010046                 8$:     MOV     R0,-(SP)        ;;MOVE CONTEXT OF REGISTER TO STACK
 21288  131334  005720                         TST     (R0)+           ;;ADVANCE POINTER
```

```
21289  131336  000401                  BR      10$             ;;GO TYPE
21290  131340  013046                  MOV     @(R0)+,-(SP)    ;;IF NOT GPR, SAVE @(R0)+ FOR TYPEOUT
21291  131342  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCI (ALL DIGITS)
21292  131344  005710                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
21293  131346  001744                  BEQ     6$              ;;BR IF NO
21294  131350  104401  131356          TYPE    ,11$            ;;TYPE TWO(2) SPACES
21295  131354  000745                  BR      7$
21296  131356  020040     000  11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
21297          131362                  .EVEN
21298
21299                                  .SBTTL  GLOBAL SUBROUTINES SECTION
21300
21301                                  ;++
21302                                  ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
21303                                  ; THAT ARE USED IN MORE THAN ONE TEST.
21304                                  ;--
21305
21306                                  ;++
21307                                  ; FUNCTIONAL DESCRIPTION:
21308                                  ;    SUBROUTINE TO INITIALIZE ALL THE MMU REGISTERS
21309
21310
21311                                  ; INPUTS: NONE
21312
21313                                  ; OUTPUTS: NONE
21314
21315                                  ; SUBORDINATE ROUTINES USED: LOAD PARS
21316                                  ;                            LOAD PDRS
21317
21318                                  ; FUNCTIONAL SIDE EFFECTS: NONE
21319
21320                                  ; CALLING SEQUENCE:     JSR     PC,INITMM
21321
21322  131362  012701  172240  INITMM: MOV     #172240,R1              ;BASE ADDRESS OF SIPARS
21323  131366  004737  131524          JSR     PC,     LDPARS
21324  131372  012701  172260          MOV     #172260,R1              ;BASE ADDRESS OF SDPARS
21325  131376  004737  131524          JSR     PC,     LDPARS
21326  131402  012701  172340          MOV     #172340,R1              ;BASE ADDRESS OF KIPARS
21327  131406  004737  131524          JSR     PC,     LDPARS
21328  131412  012701  172360          MOV     #172360,R1              ;BASE ADDRESS OF KDPARS
21329  131416  004737  131524          JSR     PC,     LDPARS
21330  131422  012701  177640          MOV     #177640,R1              ;BASE ADDRESS OF UIPARS
21331  131426  004737  131524          JSR     PC,     LDPARS
21332  131432  012701  177660          MOV     #177660,R1              ;BASE ADDRESS OF UDPARS
21333  131436  004737  131524          JSR     PC,     LDPARS
21334  131442  012701  177600          MOV     #177600,R1              ;BASE ADDRESS OF UIPDRS
21335  131446  004737  131554          JSR     PC,     LDPDRS
21336  131452  012701  177620          MOV     #177620,R1              ;BASE ADDRESS OF UDPDRS
21337  131456  004737  131554          JSR     PC,     LDPDRS
21338  131462  012701  172300          MOV     #172300,R1              ;BASE ADDRESS OF KIPDRS
21339  131466  004737  131554          JSR     PC,     LDPDRS
21340  131472  012701  172320          MOV     #172320,R1              ;BASE ADDRESS OF KDPDRS
21341  131476  004737  131554          JSR     PC,     LDPDRS
21342  131502  012701  172200          MOV     #172200,R1              ;BASE ADDRESS OF SIPDRS
21343  131506  004737  131554          JSR     PC,     LDPDRS
21344  131512  012701  172220          MOV     #172220,R1              ;BASE ADDRESS OF SDPDRS
```

```
 21345  131516  004737  131554                  JSR     PC,     LDPDRS
 21346  131522  000207                          RTS     PC                      ;RETURN
```

```
21347                                   ;++
21348                                   ; FUNCTIONAL DESCRIPTION:
21349                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE ADDRESS REGISTERS (PARS).
21350                                   ;       THIS ROUTINE WILL INITIALIZE 8 PARS STARTING AT A BASE ADDRESS
21351                                   ;       SUPPLIED BY THE CALLING ROUTINE. PARS 0-5 WILL BE MAPPED FROM
21352                                   ;       ADDRESS 0 TO ADDRESS 137777 (0-24K). PAR 6 WILL BE MAPPED FROM
21353                                   ;       ADDRESS 200000 TO 217777 AND PAR 7 WILL BE MAPPED TO THE I/O
21354                                   ;       PAGE.
21355                                   ;
21356                                   ; INPUTS:
21357                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PARS TO BE INITIALIZED
21358                                   ;
21359                                   ; OUTPUTS: NONE
21360                                   ;
21361                                   ; SUBORDINATE ROUTINES USED: NONE
21362                                   ;
21363                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21364                                   ;
21365                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21366                                   ;
21367  131524  012702  000006        LDPARS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21368  131530  005003                        CLR     R3                      ;INITIALIZE INDEX VALUE
21369  131532  010321                1$:     MOV     R3,     (R1)+           ;LOAD PARS
21370  131534  062703  000200              ADD     #200,   R3              ;INDEX IN 4K INCREMENTS
21371  131540  077204                        SOB     R2,     1$              ;LOAD FIRST SIX PARS
21372  131542  012721  002000              MOV     #2000,  (R1)+           ;LET PAR6 MAP TO 200000
21373  131546  012711  177600              MOV     #177600,(R1)            ;LET PAR7 MAP TO I/O PAGE
21374  131552  000207                        RTS     PC                      ;RETURN
```

```
21375                                   ;++
21376                                   ; FUNCTIONAL DESCRIPTION:
21377                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE DECRIPTOR REGISTERS (PDRS).
21378                                   ;       THIS ROUTINE WILL INITIALIZE 8 PDRS STARTING AT A BASE ADDRESS
21379                                   ;       SUPPLIED BY THE CALLING ROUTINE. PDRS 0-5 WILL BE INITIALIZED TO
21380                                   ;       4K READ/WRITE BYPASS AND PDRS 6 AND 7 WILL BE INITIALIZED TO
21381                                   ;       4K READ/WRITE NO BYPASS.
21382                                   ;       NOTE:   THERE IS NO NEED TO BYPASS ON I/O PAGE REFERENCES BECAUSE
21383                                   ;               THE CACHE DOES NOT ALLOCATE ANY OF THESE REFERENCES.
21384
21385                                   ; INPUTS:
21386                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PDRS TO BE INITIALIZED
21387
21388                                   ; OUTPUTS: NONE
21389
21390                                   ; SUBORDINATE ROUTINES USED: NONE
21391
21392                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21393
21394                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21395
21396  131554  012702  000006          LDPDRS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21397  131560  012721  177406          1$:     MOV     #177406,(R1)+           ;LOAD PDRS WITH 4K READ/WRITE BYPASS
21398  131564  077203                          SOB     R2,     1$              ;LOAD FIRST SIX PDRS
21399  131566  012721  077406                  MOV     #77406, (R1)+           ;LET PAR6 BE 4K READ/WRITE NO BYPASS
21400  131572  012711  077406                  MOV     #77406, (R1)            ;LET PAR7 BE 4K READ/WRITE NO BYPASS ALSO
21401  131576  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)
                                        ;
131600  011637  001122          RAMPAR: MOV     (SP),$BDADR             ;STOR ADDESS TRAPPED
131604  032737  000100  177744          BIT     #BIT06, MSER            ;IF LOW BYTE PARITY ERROR
131612  001401                          BEQ     1$                      ;THEN
131614  104007                          ERROR   .+7                     ;ERROR
131616  032737  000200  177744  1$:     BIT     #BIT07, MSER            ;IF HIGH BYTE PARITY ERROR
131624  001401                          BEQ     2$                      ;THEN
131626  104007                          ERROR   .+7                     ;ERROR
131630  032737  000040  177744  2$:     BIT     #BIT05, MSER            ;IF TAG PARITY ERROR
131636  001401                          BEQ     3$                      ;THEN
131640  104007                          ERROR   .+7                     ;ERROR
131642  005037  177744          3$:     CLR     MSER                    ;INITIALIZE MSER AFTER ERROR
131646  000002                          RTI                             ;RETURN
131650  005237  002702          LKSINT: INC     LKSFL                   ;INCREMENT FLAG
131654  000002                          RTI
                                        ;
                                        .SBTTL Q22BE SIZE ROUTINE
                                        ;THIS ROUTINE WILL AUTOSIZE FOR UP TO TWO Q22 BUS EXERCISERS.  IF NONE
                                        ;FOUND LOCATIONS CSR1 AND CSR12 WILL BE LEFT ZEROES.  THIS ROUTINE WILL
                                        ;ONLY RUN IN NOT UFD MODE.
                                        ;
131656  032737  001000  177750  Q22SIZ: BIT     #BIT09,MAIREG                   ;UNIBUS SYSTEM?
131664  001401                          BEQ     1$                              ;IF NOT, ADVANCE TO ROUTINE
131666  000207                          RTS     PC                              ;OTHERWISE, RETURN
                                        ;
                                        ; PREPARE TO DO SIZING
                                        ;
131670  013701  000004          1$:     MOV     ERRVEC,R1                       ;STORE TIMEOUT VECTOR
131674  012737  132050  000004          MOV     #7$,ERRVEC                      ;POINT NEW TO PROGRAM
131702  012737  000340  000006          MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
131710  005037  001160                  CLR     $TMP0                           ;CLEAR Q22BE COUNTER
131714  012702  170000                  MOV     #170000,R2                      ;FIRST POSSIBLE ADDRESS
131720  012703  000510                  MOV     #510,R3                         ;VECTOR FOR IT
131724  000404                          BR      3$                              ;TRY THOSE VALUES
                                        ;
                                        ; NOW DO ACTUAL SIZING
                                        ;
131726  062702  000020          2$:     ADD     #20,R2                          ;GET CSR FOR NEXT Q22BE
```

```
21458  131732  062703  000004                    ADD     #4,R3                   ;GET VECTOR FOR NEXT ONE
21459  131736  005712                  3$:       TST     (R2)                    ;TRY TO ACCESS CSR
21460                                             ;
21461                                             ; IF NO TIMEOUT, STORE EXISTING ADDRESSES TO REGISTERS
21462                                             ;
21463  131740  005737  001160                    TST     $TMP0                   ;FIRST Q22BE FOUND?
21464  131744  001010                            BNE     4$                      ;IF SECOND, BRANCH
21465  131746  012705  002644                    MOV     #CSR1,R5                ;START WITH CSR1 FOR 1ST
21466  131752  010237  002662                    MOV     R2,SIMGOA               ;SIMULTANEOUS GO
21467  131756  062737  000016  002662            ADD     #16,SIMGOA              ;ADDRESS
21468  131764  000402                            BR      5$                      ;BRANCH TO INITIALISE
21469  131766  012705  002664          4$:       MOV     #CSR12,R5               ;START WITH CSR12 FOR 2ND
21470  131772  012704  000004          5$:       MOV     #4,R4                   ;INITIALISE 5 REGISTERS
21471  131776  010215                            MOV     R2,(R5)                 ;INITIALISE CSR1
21472  132000  011565  000002          6$:       MOV     (R5),2(R5)              ;STORE TO NEXT ONE
21473  132004  005725                            TST     (R5)+                   ;GET NEXT ADDRESS
21474  132006  062715  000002                    ADD     #2,(R5)                 ;GET ADDRESS, POINT NEXT
21475  132012  077406                            SOB     R4,6$                   ;DO FOR NEXT 4 REGISTERS
21476  132014  010365  000002                    MOV     R3,2(R5)                ;STORE INTERRUPT VECTOR
21477  132020  010365  000004                    MOV     R3,4(R5)                ;AND PRIORITY
21478  132024  062765  000002  000004            ADD     #2,4(R5)
21479  132032  005237  001160                    INC     $TMP0                   ;COUNT Q22BE'S
21480  132036  022737  000002  001160            CMP     #2,$TMP0                ;TWO FOUND?
21481  132044  001406                            BEQ     9$                      ;IF SO, STOP SIZING
21482  132046  000402                            BR      8$                      ;OTHERWISE, CONTINUE SIZING
21483                                             ;
21484                                             ; ON TIMEOUT TRY TO LOOK AT NEXT ADDRESS RANGE
21485                                             ;
21486  132050  005726                  7$:       TST     (SP)+                   ;RESTORE STACK FROM
21487  132052  005726                            TST     (SP)+                   ;TIMEOUT
21488  132054  022702  170160          8$:       CMP     #170160,R2              ;AT THE LAST POSSIBLE?
21489  132060  001322                            BNE     2$                      ;IF NOT, BRANCH
21490  132062  005737  002644          9$:       TST     CSR1                    ;1 FOUND?
21491  132066  001002                            BNE     10$                     ;IF YES, BRANCH
21492  132070  104401  132102                    TYPE    ,NOQ22                  ;TYPE NO FOUND
21493  132074  010137  000004          10$:      MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
21494  132100  000207                            RTS     PC                      ;RETURN
21495
21496  132102  006412  047516  050440  NOQ22:   .ASCIZ  <12><15>/NO Q22BE FOUND IN THE SYSTEM/
21497  132110  031062  042502  043040
21498  132116  052517  042116  044440
21499  132124  020116  044124  020105
21500  132132  054523  052123  046505
21501  132140     000
21502          132142                           .EVEN
21503                                           .SBTTL Q22BE INTERRUPT INITIALISE ROUTINE
21504                                           ;THIS ROUTINE WILL INITIALISE Q22BE TO INTERRUPT AT A PRIORITY AT (R0)+
21505                                           ;AT THE STARTING ADDRESS IN R3. THE TEST HAVE TO SET ACTUAL DONE BIT
21506                                           ;BY CLEARING GO.
21507
21508  132142  005013                  Q22INT:   CLR     (R3)                    ;CLEAR TRANSFER TYPE IN CSR1
21509  132144  052710  000001                    BIS     #BIT00,(R0)             ;ZERO DONE
21510  132150  011063  000002                    MOV     (R0),2(R3)              ;SET PRIORITY IN CSR2
21511  132154  042710  000001                    BIC     #BIT00,(R0)             ;PREPARE TO SET DONE
21512  132160  000207                            RTS     PC
21513
```

```
21514                                   .SBTTL DMATRN DATO CYCLE THRU Q22BE
21515                                   ;THIS ROUTINE PERFORMS DATO FROM A LOCATION TEMP THRU THE FIRST
21516                                   ;FOUND Q22BE STARTING AT LOCATION @CSR1. R0 HAS 0 IF ONLY 1 TRANSFER IS
21517                                   ;TO BE PERFORMED. OTHERWISE 16 BLOCK MODE TRANSFERS ARE TO BE PERFORMED.
21518                                   ;IN THE LATTER CASE ADDRESS AND WORD COUNT HAS TO BE LOADED BEFORE.
21519
21520  132162  012777  012525  050464  DMATRN: MOV     #12525,@DATA                    ;DATA USED
21521  132170  005700                          TST     R0                              ;DO 1 WORD?
21522  132172  001404                          BEQ     1$                              ;IF YES, BRANCH
21523  132174  012777  001001  050444          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE, GO
21524  132202  000414                          BR      2$                              ;BRANCH TO DO IT
21525  132204  012777  001601  050432  1$:     MOV     #1601,@CSR1                     ;RESET LATENCY COUNT,DATO
21526  132212  012777  002710  050430          MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21527  132220  012777  177777  050424          MOV     #177777,@WC                     ;DO 1 WORD
21528  132226  012777  000001  050412          MOV     #BIT00,@CSR2                    ;DO IT
21529  132234  105777  050406          2$:     TSTB    @CSR2                           ;DMA DONE?
21530  132240  100375                          BPL     2$                              ;WAIT TILL DONE
21531  132242  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21532
21533                                   .SBTTL DMARD DATI THRU Q22BE
21534                                   ;THIS ROUTINE PERFORMS DATI CYCLE THRU Q22BE IN EITHER BLOCK MODE OR A SINGLE
21535                                   ;TRANSFER MODE. MEMORY LOCATION USED IS TEMP. R0 IS ZERO FOR SINGLE TRANSFER
21536
21537  132244  012777  002710  050376  DMARD:  MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21538  132252  005700                          TST     R0                              ;DO 1 WORD?
21539  132254  001412                          BEQ     1$                              ;IF YES, BRANCH
21540  132256  012777  001507  050360          MOV     #1507,@CSR1                     ;16 DATIB
21541  132264  012777  177770  050360          MOV     #177770,@WC                     ;DO 8 WORD
21542  132272  012777  001001  050346          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE GO
21543  132300  000411                          BR      2$                              ;GO CHECK
21544  132302  012777  001407  050334  1$:     MOV     #1407,@CSR1                     ;RESET LATENCY COUNT,DATI
21545                                                                                  ;LOAD NEW DATA TO DATA R.
21546  132310  012777  177777  050334          MOV     #177777,@WC                     ;DO 1 WORD
21547  132316  012777  000001  050322          MOV     #BIT00,@CSR2                    ;DO IT
21548  132324  105777  050316          2$:     TSTB    @CSR2                           ;DMA DONE?
21549  132330  100375                          BPL     2$                              ;WAIT TILL DONE
21550  132332  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21551
21552
21553
21554  132334  011637  001122          TOUT:   MOV     (SP),@BDADR             ;STORE TRAPPED PC
21555  132340  104130                          ERROR   .130                    ;UNEXPECTED TRAP
21556  132342  000002                          RTI
21557
21558                                   ;
21559                                   ;MMU GLOBAL SUBROUTINES
21560                                   ;
21561                                   ;
21562                                   ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT
21563                                   ;
21564  132344  010046                  MMU:    MOV     R0,-(SP)                ;SAVE CONTENTS OF REGISTERS
21565  132346  010146                          MOV     R1,-(SP)                ;
21566  132350  010246                          MOV     R2,-(SP)                ;
21567  132352  012700  177600                  MOV     #177600,R0              ;
21568  132356  004737  132444                  JSR     PC,PDR                  ;INIT I AND D USER PDR'S
21569  132362  004737  132466                  JSR     PC,PAR                  ;INIT I USER PAR'S
```

```
21570  132366  004737  132466                  JSR     PC,PAR          ;INIT D USER PAR'S
21571  132372  012700  172200                  MOV     #172200,R0
21572  132376  004737  132444                  JSR     PC,PDR          ;INIT I AND D SUP PDR'S
21573  132402  004737  132466                  JSR     PC,PAR          ;INIT I SUP PAR'S
21574  132406  004737  132466                  JSR     PC,PAR          ;INIT D SUP PAR'S
21575  132412  004737  132444                  JSR     PC,PDR          ;INIT I AND D KER PDR'S
21576  132416  004737  132466                  JSR     PC,PAR          ;INIT I KER PAR'S
21577  132422  004737  132466                  JSR     PC,PAR          ;INIT D KER PAR'S
21578  132426  012737  000027  172516          MOV     #27,@#172516    ;INIT MMR3
21579  132434  012602                          MOV     (SP)+,R2        ;RESTORE REGISTERS
21580  132436  012601                          MOV     (SP)+,R1        ;
21581  132440  012600                          MOV     (SP)+,R0        ;
21582  132442  000207                          RTS     PC              ;RETURN
21583                                  ;
21584                                  ;ROUTINE TO INITIALIZE PDR'S
21585                                  ;
21586  132444  005002                  PDR:    CLR     R2              ;INIT CNTR
21587  132446  012720  077406          PDR1:   MOV     #77406,(R0)+    ;INIT PDR
21588  132452  062702  000001                  ADD     #1,R2           ;INCREMENT CNTR
21589  132456  022702  000020                  CMP     #16.,R2         ;ARE WE DONE?
21590  132462  001371                          BNE     PDR1            ;BRANCH IF NOT
21591  132464  000207                          RTS     PC              ;RETURN
21592                                  ;
21593                                  ;ROUTINE TO INITIALIZE PAR'S
21594                                  ;
21595  132466  005001                  PAR:    CLR     R1              ;SETUP TO INIT PAR
21596  132470  010120                  PAR1:   MOV     R1,(R0)+        ;INIT PAR
21597  132472  062701  000200                  ADD     #200,R1         ;GET READY FOR NEXT PAR
21598  132476  022701  001600                  CMP     #1600,R1        ;REACHED A PAR7
21599  132502  001372                          BNE     PAR1            ;BRANCH IF NOT
21600  132504  012720  177600                  MOV     #177600,(R0)+   ;INIT PAR7
21601  132510  000207                          RTS     PC              ;RETURN
21602                                  ;
21603                                  ;TIME OUT ROUTINE
21604                                  ;
21605  132512  005205                  ADDTRP: INC     R5              ;INCREMENT TIME OUT FLAG
21606  132514  000002                          RTI                     ;RETURN
21607                                  ;
21608                                  ;MMU TRAP ROUTINE
21609                                  ;
21610  132516  023727  002776  000001  MMUTRP: CMP     FLAG,#1         ;ARE WE EXPECTING AN ABORT
21611  132524  001401                          BEQ     1$              ;YES GO ON
21612  132526  104002                          ERROR   .2              ;NO GO TO ERROR
21613  132530  010046                  1$:     MOV     R0,-(SP)        ;SAVE CONTENTS OF REG 0
21614  132532  013700  177776                  MOV     @#177776,R0     ;SAVE A COPY OF PSW
21615  132536  072027  177764                  ASH     #-14,R0         ;LOOK AT BITS<15:14>
21616  132542  020027  000002                  CMP     R0,#2           ;WAS PS<15:14>=10
21617  132546  001001                          BNE     OK              ;NO GO ON
21618  132550  000411                          BR      NOTOK           ;YES CHANGE BITS TO 00
21619  132552  013700  177776          OK:     MOV     @#177776,R0     ;SAVE A COPY OF PSW
21620  132556  072027  000002                  ASH     #2,R0           ;LOOK AT BITS<13:12>
21621  132562  072027  177764                  ASH     #-14,R0         ;
21622  132566  020027  000002                  CMP     R0,#2           ;WAS PS<13:12>=10
21623  132572  001002                          BNE     OK1             ;NO GO ON
21624  132574  005066  000004          NOTOK:  CLR     4(SP)           ;CLEAR ILLEGAL MODE FFROM OLD PSW
21625  132600  013737  177572  003004  OK1:    MOV     @#177572,SAVMR0 ;SAVE A COPY OF MMR0
```

E1


```
 21626   132606   013737   177574   003006          MOV     @#177574,SAVMR1         ;SAVE A COPY OF MMR1
 21627   132614   013737   177576   003010          MOV     @#177576,SAVMR2         ;SAVE A COPY OF MMR2
 21628   132622   005037   177572                   CLR     @#177572                ;CLEAR ABORT BITS AND TURN MMU OFF
 21629   132626   005037   002776                   CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 21630   132632   012600                            MOV     (SP)+,R0                ;RESTORE ORIGINAL CONTENTS OF REG 0
 21631   132634   000002                            RTI                             ;RETURN
```

```
 21632                                          ;FPP COMMON SUBROUTINES
 21633  132636  012600                         WLDTRP: MOV     (SP)+,R0                ;SAVE PC
 21634  132640  012605                                 MOV     (SP)+,R5                ;SAVE STATUS AND RESTORE STACK
 21635  132642  104003                                 ERROR   +3
 21636  132644  000110                                 JMP     (R0)                    ;GO BACK INLINE
 21637                                          ;
 21638                                          ;
 21639                                          ;
 21640  132646  000000                         TRPFLG: .WORD   0
 21641  132650  000207                         ERRFP:  RTS     R7
 21642  132652  000207                         ERR:    RTS     R7
 21643                                          ;
 21644                                          ;
 21645                                          ;
 21646                                          ;
 21647                                          ;
 21648                                          ;SUBROUTINE DATA VERFICATION -
 21649                                          ;
 21650                                          ; CALLED BY     JSR     R7,DATVER
 21651                                          ;
 21652                                          ;INPUT:         (R4)=EXPECTED DATA
 21653                                          ;               (R1)=RECEIVED DATA
 21654                                          ;
 21655                                          ;THIS ROUTINE VERIFIES THAT THE 4 CONSECTIVE WORDS STARTING WITH (R4) ARE
 21656                                          ;EQUAL TO THE FOUR WORDS ADDRESSED BY (R1). THE CONTENTS OF R4, AND R1 ARE NOT
 21657                                          ;DISTURBED.
 21658                                          ;LOCATION "COUNT" , IF NOT EQUAL TO 0 SIGNIFIES DATA ERROR
 21659                                          ;IF THE STATUS IS FLOATING MODE, THE LAST TWO BYTES OF RECEIEVED
 21660                                          ;ARE SIMPLY CHECKED FOR ZEROS
 21661                                          ;
 21662
 21663  132654  010446                         DATVFR: MOV     R4,-(SP)                ;SAVE R4
 21664  132656  010146                                 MOV     R1,-(SP)                        ;SAVE R1
 21665  132660  012737  000003  003060                 MOV     #3,COUNT                ;SET UP ITERATION COUNT
 21666  132666  000137  132704                         JMP     DAT1                    ;
 21667                                          ;
 21668  132672  010446                         DATVER: MOV     R4,-(SP)                ;SAVE R4
 21669  132674  010146                                 MOV     R1,-(SP)                ;SAVE R1
 21670  132676  012737  000005  003060                 MOV     #5,COUNT                ;SET UP ITERATION COUNT
 21671  132704  005337  003060                 DAT1:   DEC     COUNT
 21672  132710  001402                                 BEQ     2$                      ;BRANCH IF DONE
 21673  132712  022421                                 CMP     (R4)+,(R1)+                     ;
 21674  132714  001773                                 BEQ     DAT1
 21675  132716  012601                         2$:     MOV     (SP)+,R1                ;RESTORE R1
 21676  132720  012604                                 MOV     (SP)+,R4                ;RESTORE R4
 21677  132722  000207                                 RTS     R7                      ;GO BACK TO CALLING ROUTINE
 21678                                                                                  ;IF DATA ERROR, COUNT NE 0
```

```
21679
21680
21681                                   .SBTTL  END OF PASS ROUTINE
21682
21683                                   ;;**********************************************************************
21684                                   ;*INCREMENT THE PASS NUMBER ($PASS)
21685                                   ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
21686                                   ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
21687                                   ;*IF THERES A MONITOR GO TO IT
21688                                   ;*IF THERE ISN'T JUMP TO LOOP
21689
21690  132724                           $EOP:
21691  132724  032737  000040  000052          BIT     #BIT05,@#52
21692  132732  001026                        BNE     $GET42
21693  132734  005037  001102                CLR     $TSTNM          ;;ZERO THE TEST NUMBER
21694  132740  005037  001164                CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
21695  132744  005237  001206                INC     $PASS           ;;INCREMENT THE PASS NUMBER
21696  132750  042737  100000  001206        BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
21697  132756  005327                        DEC     (PC)+           ;;LOOP?
21698  132760  000001                 $EOPCT: .WORD   1
21699  132762  003022                        BGT     $DOAGN          ;;YES
21700  132764  012737                        MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
21701  132766  000001                 $ENDCT: .WORD   1
21702  132770  132760                        $EOPCT
21703  132772  104401  133037                TYPE    ,$ENDMG         ;;TYPE "END PASS #"
21704  132776  013746  001206                MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
21705  133002  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
21706  133004  104401  133034                TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
21707  133010  013700  000042         $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
21708  133014  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
21709  133016  000005                        RESET                   ;;CLEAR THE WORLD
21710  133020  004710                 $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
21711  133022  000240                        NOP                     ;;SAVE ROOM
21712  133024  000240                        NOP                     ;;FOR
21713  133026  000240                        NOP                     ;;ACT11
21714  133030                         $DOAGN:
21715  133030  000137                        JMP     @(PC)+          ;;RETURN
21716  133032  004534                 $RTNAD: .WORD   LOOP
21717  133034     377        377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
21718  133037     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
21719  133044  050040  051501  020123
21720  133052  000043
21721
21722                                   .SBTTL  SCOPE HANDLER ROUTINE
21723
21724                                   ;;**********************************************************************
21725                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
21726                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
21727                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
21728                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
21729                                   ;*SW14=1         LOOP ON TEST
21730                                   ;*SW11=1         INHIBIT ITERATIONS
21731                                   ;*SW09=1         LOOP ON ERROR
21732                                   ;*SW08=1         LOOP ON TEST IN SWR<5:0>
21733                                   ;*CALL
21734                                   ;*      SCOPE           ;;SCOPE=IOT
```

```
21735  133054                                 $SCOPE:
21736  133054  104407                                 CKSWR                          ;;TEST  FOR CHANGE IN SOFT-SWR
21737  133056  032777  040000  046054  1$:     BIT     @#BIT14,@SWR                   ;;LOOP ON PRESENT TEST?
21738  133064  001117                         BNE     $OVER                          ;;YES IF SW14=1
21739                                         ;*****START OF CODE FOR THE XOR TESTER*****
21740  133066  000416                         $XTSTR: BR      6$                     ;;IF RUNNING ON THE "XOR" TESTER CHANGE
21741                                                                                 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
21742  133070  013746  000004                         MOV     @#ERRVEC,-(SP)         ;;SAVE THE CONTENTS OF THE ERROR VECTOR
21743  133074  012737  133114  000004               MOV     #5$,@#ERRVEC           ;;SET FOR TIMEOUT
21744  133102  005737  177060                       TST     @#177060               ;;TIME OUT ON XOR?
21745  133106  012637  000004                       MOV     (SP)+,@#ERRVEC         ;;RESTORE THE ERROR VECTOR
21746  133112  000466                               BR      $SVLAD                 ;;GO TO THE NEXT TEST
21747  133114  022626                         5$:     CMP     (SP)+,(SP)+            ;;CLEAR THE STACK AFTER A TIME OUT
21748  133116  012637  000004                       MOV     (SP)+,@#ERRVEC         ;;RESTORE THE ERROR VECTOR
21749  133122  000426                               BR      7$                     ;;LOOP ON THE PRESENT TEST
21750  133124                                 6$:;*****END OF CODE FOR THE XOR TESTER*****
21751  133124  032777  000400  046006               BIT     #BIT08,@SWR            ;;LOOP ON SPEC. TEST?
21752  133132  001407                               BEQ     2$                     ;;BR IF NO
21753  133134  017746  046000                       MOV     @SWR,-(SP)             ;;SET DESIRED TEST NUM. FROM SWR
21754  133140  042716  000300                       BIC     #$SWRMK,(SP)           ;;STRIP AWAY UNDESIRED BITS
21755  133144  122637  001102                       CMPB    (SP)+,$TSTNM           ;;ON THE RIGHT TEST?
21756  133150  001465                               BEQ     $OVER                  ;;BR IF YES
21757  133152  105737  001103                 2$:     TSTB    $ERFLG                 ;;HAS AN ERROR OCCURRED?
21758  133156  001421                               BEQ     3$                     ;;BR IF NO
21759  133160  123737  001115  001103               CMPB    $ERMAX,$ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
21760  133166  101015                               BHI     3$                     ;;BR IF NO
21761  133170  032777  001000  045742               BIT     #BIT09,@SWR            ;;LOOP ON ERROR?
21762  133176  001404                               BEQ     4$                     ;;BR IF NO
21763  133200  013737  001110  001106  7$:     MOV     $LPERR,$LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
21764  133206  000446                               BR      $OVER
21765  133210  105037  001103                 4$:     CLRB    $ERFLG                 ;;ZERO THE ERROR FLAG
21766  133214  005037  001164                       CLR     $TIMES                 ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
21767  133220  000415                               BR      1$                     ;;ESCAPE TO THE NEXT TEST
21768  133222  032777  004000  045710  3$:     BIT     #BIT11,@SWR            ;;INHIBIT ITERATIONS?
21769  133230  001011                               BNE     1$                     ;;BR IF YES
21770  133232  005737  001206                       TST     $PASS                  ;;IF FIRST PASS OF PROGRAM
21771  133236  001406                               BEQ     1$                     ;;      INHIBIT ITERATIONS
21772  133240  005237  001104                       INC     $ICNT                  ;;INCREMENT ITERATION COUNT
21773  133244  023737  001164  001104               CMP     $TIMES,$ICNT           ;;CHECK THE NUMBER OF ITERATIONS MADE
21774  133252  002024                               BGE     $OVER                  ;;BR IF MORE ITERATION REQUIRED
21775  133254  012737  000001  001104  1$:     MOV     #1,$ICNT               ;;REINITIALIZE THE ITERATION COUNTER
21776  133262  013737  133340  001164               MOV     $MXCNT,$TIMES          ;;SET NUMBER OF ITERATIONS TO DO
21777  133270  105237  001102                 $SVLAD: INCB    $TSTNM                 ;;COUNT TEST NUMBERS
21778  133274  113737  001102  001204               MOVB    $TSTNM,$TESTN          ;;SET TEST NUMBER IN APT MAILBOX
21779  133302  011637  001106                       MOV     (SP),$LPADR            ;;SAVE SCOPE LOOP ADDRESS
21780  133306  011637  001110                       MOV     (SP),$LPERR            ;;SAVE ERROR LOOP ADDRESS
21781  133312  005037  001164                       CLR     $ESCAPE                ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
21782  133316  112737  000001  001115               MOVB    #1,$ERMAX              ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
21783  133324  013777  001102  045610  $OVER:  MOV     $T_TNM,@DISPLAY        ;;DISPLAY TEST NUMBER
21784  133332  013716  001106                       MOV     $LPADR,(SP)            ;;FUDGE RETURN ADDRESS
21785  133336  000002                               RTI                            ;;FIXES PS
21786  133340  000001                         $MXCNT: 1                              ;;MAX. NUMBER OF ITERATIONS
21787                                         .SBTTL  ERROR HANDLER ROUTINE
21788
21789                                         ;;**************************************************************
21790                                         ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

```
21791                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
21792                                  ;*AND GO TO ERTYPE ON ERROR
21793                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
21794                                  ;*SW15=1        HALT ON ERROR
21795                                  ;*SW13=1        INHIBIT ERROR TYPEOUTS
21796                                  ;*SW10=1        BELL ON ERROR
21797                                  ;*SW09=1        LOOP ON ERROR
21798                                  ;*CALL
21799                                  ;*      ERROR   *N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
21800
21801  133342                          $ERROR:
21802  133342  005737  004060                  TST     UQUIET          ;;TEST FOR USER-QUIET MODE
21803  133346  001403                          BEQ     9$              ;;BRANCH IF FIELD-SERVICE MODE
21804  133350  005000                          CLR     R0              ;;IN CASE R0 HAS A #3 IN IT (^C)
21805  133352  004737  133556                  JSR     PC,ABORT        ;;TEST FOR ABORT CONDITION
21806  133356                          9$:
21807  133356  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
21808  133360  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
21809  133364  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
21810  133366  013777  001102  045546          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
21811  133374  032777  002000  045536          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
21812  133402  001402                          BEQ     1$              ;;NO - SKIP
21813  133404  104401  001170                  TYPE    .$BELL          ;;RING BELL
21814  133410  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
21815  133414  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
21816  133420  162737  000002  001116          SUB     #2,$ERRPC
21817  133426  117737  045464  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
21818  133434  032777  020000  045476          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
21819  133442  001004                          BNE     20$             ;;SKIP TYPEOUTS
21820  133444  004737  131144                  JSR     PC,ERTYPE       ;;GO TO USER ERROR ROUTINE
21821  133450  104401  001175                  TYPE    .$CRLF
21822  133454                          20$:
21823  133454  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
21824  133462  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
21825  133464  113737  001114  133476          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
21826  133472  004737  133734                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
21827  133476     000          21$:    .BYTE   0
21828  133477     000                  .BYTE   0
21829  133500  000777          22$:    BR      22$             ;;APT ERROR LOOP
21830  133502  005777  045432          2$:     TST     @SWR            ;;HALT ON ERROR
21831  133506  100002                          BPL     3$              ;;SKIP IF CONTINUE
21832  133510  000000                          HALT                    ;;HALT ON ERROR!
21833  133512  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
21834  133514  032777  001000  045416  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
21835  133522  001402                          BEQ     4$              ;;BR IF NO
21836  133524  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
21837  133530  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
21838  133534  001402                          BEQ     5$              ;;BR IF NONE
21839  133536  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
21840  133542                          5$:
21841  133542  022737  133020  000042          CMP     #$ENCAD,@#42    ;;ACT-11 AUTO-ACCEPT?
21842  133550  001001                          BNE     6$              ;;BRANCH IF NO
21843  133552  000000                          HALT                    ;;YES
21844  133554                          6$:
21845  133554  000002                          RTI                     ;;RETURN
21846
```

```
21847
21848
21849                                   .SBTTL  ABORT ROUTINE FOR LCP/ORION UFD MODE
21850
21851
21852  133556  005737  004056          ABORT:  TST     UFDFLG          ;TEST FOR USER FRIENDLY MODE
21853  133562  001454                          BEQ     NOABRT          ;IF NOT UFD THEN CONTINUE NORMAL OPERATION
21854  133564  020027  000032                  CMP     R0,#32          ;IS IT A ↑Z ?
21855  133570  001443                          BEQ     ABORTZ          ;JUST GO BACK TO CHAIN IF IT IS (NO ERROR)
21856  133572  020027  000003                  CMP     R0,#3           ;IS IS A ↑C ?
21857  133576  001404                          BEQ     ABORTC          ;BR TO LOAD ↑C ON XXDP+ STACK (NO ERROR)
21858  133600  005737  004060                  TST     UQUIET          ;TEST FOR USER-QUIET MODE
21859  133604  001443                          BEQ     NOABRT          ;IF FIELD-SERVICE MODE, CONTINUE NORMAL OPERATION
21860                                                                  ; BECAUSE FIELD-SERVICE MODE DOES NOT QUIT ON ERROR
21861  133606  000422                          BR      ABORTE          ;SET DRSERR THEN LEAVE
21862
21863
21864
21865  133610  013737  004052  000030  ABORTC: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
21866  133616  013737  004054  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
21867  133624  104043                          EMT     +43             ;GET XXDP STACK LOC. INTO R0 FROM MONITOR
21868  133626  005720                  1$:     TST     (R0)+           ;FIND END OF STACK
21869  133630  001376                          BNE     1$
21870  133632  112760  000057  177777          MOVB    #'/,-1(R0)      ;LOAD SLASH OVER ZERO
21871  133640  112720  000136                  MOVB    #'↑,(R0)+       ;LOAD UPARROW
21872  133644  112720  000103                  MOVB    #'C,(R0)+       ;LOAD C
21873  133650  105010                          CLRB    (R0)            ;MAKE NEW END TO STACK
21874  133652  000412                          BR      ABORTZ          ;NOW LEAVE
21875  133654  013737  004052  000030  ABORTE: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
21876  133662  013737  004054  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
21877  133670  104042                          EMT     +42             ;GET DCA LOCATION INTO R0 FROM MONITOR
21878  133672  012760  177777  000042          MOV     #-1,42(R0)      ;SET A -1 INTO LOCATION DRSERR IN MONITOR
21879  133700  013700  000042  ABORTZ: MOV     @#42,R0         ;AND PUT THE MONITOR RETURN ADDRESS IN R0
21880  133704  005037  000042          CLR     @#42            ;CLEAR MONITOR RETURN FLAG
21881  133710  000137  133020          JMP     $ENDAD          ;RETURN TO MONITOR-DO NOT PUSH STACK HERE
21882  133714  000207                  NOABRT: RTS     PC              ;IF NOTUFD RETURN TO MAINLINE
21883
21884
21885                                   .SBTTL  APT COMMUNICATIONS ROUTINE
21886
21887                                   ;;**************************************************************************
21888  133716  112737  000001  134162  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
21889  133724  112737  000001  134160  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
21890  133732  000403                          BR      $ATYC
21891  133734  112737  000001  134162  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
21892  133742                          $ATYC:
21893  133742  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
21894  133744  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
21895  133746  105737  134160                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
21896  133752  001450                          BEQ     5$              ;;IF NOT:  BR
21897  133754  122737  000001  001220          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
21898  133762  001031                          BNE     3$              ;;IF NOT:  BR
21899  133764  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
21900  133772  001425                          BEQ     3$              ;;IF NOT:  BR
21901  133774  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
21902  134000  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
```

```
21903  134006  005737  001200          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
21904  134012  001375                          BNE     1$              ;;IF NOT:  WAIT
21905  134014  010037  001214                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
21906  134020  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
21907  134022  001376                          BNE     2$
21908  134024  163700  001214                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
21909  134030  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
21910  134032  010037  001216                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
21911  134036  012737  000004  001200          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
21912  134044  000413                          BR      5$
21913  134046  017637  000004  134072  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
21914  134054  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
21915  134062  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
21916  134066  004737  134164                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
21917  134072  000000                  4$:     .WORD   0
21918  134074                          5$:
21919  134074  105737  134162          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
21920  134100  001416                          BEQ     12$             ;;IF NOT:  BR
21921  134102  005737  001220                  TST     $ENV            ;;RUNNING UNDER APT?
21922  134106  001413                          BEQ     12$             ;;IF NOT:  BR
21923  134110  005737  001200          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
21924  134114  001375                          BNE     11$             ;;IF NOT:  WAIT
21925  134116  017637  000004  001202          MOV     @4(SP),$FATAL   ;;GET ERROR #
21926  134124  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
21927  134132  005237  001200                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
21928  134136  105037  134162          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
21929  134142  105037  134161                  CLRB    $LFLG           ;;CLEAR LOG FLAG
21930  134146  105037  134160                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
21931  134152  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
21932  134154  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
21933  134156  000207                          RTS     PC              ;;RETURN
21934  134160     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
21935  134161     000          $LFLG:  .BYTE   0               ;;LOG FLAG
21936  134162     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
21937          134164                  .EVEN
21938          000200          APTSIZE=200
21939          000001          APTENV=001
21940          000100          APTSPOOL=100
21941          000040          APTCSUP=040
21942                          .SBTTL  TYPE ROUTINE
21943
21944                          ;;**************************************************************
21945                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
21946                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
21947                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
21948                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
21949                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
21950                          ;*
21951                          ;*CALL:
21952                          ;*1) USING A TRAP INSTRUCTION
21953                          ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
21954                          ;*OR
21955                          ;*      TYPE
21956                          ;*      MESADR
21957                          ;*
21958
```

```
21959  134164  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
21960  134170  100002                          BPL     1$              ;;BR IF YES
21961  134172  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
21962  134174  000430                          BR      3$              ;;LEAVE
21963  134176  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
21964  134200  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
21965  134204  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
21966  134212  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
21967  134214  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
21968  134222  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
21969  134224  010037  134234                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
21970  134230  004737  133724                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
21971  134234  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
21972  134236  132737  000040  001221  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
21973  134244  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
21974  134246  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
21975  134250  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
21976  134252  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
21977  134254  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
21978  134256  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
21979  134262  000002                          RTI                     ;;RETURN
21980  134264  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
21981  134270  001430                          BEQ     8$
21982  134272  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
21983  134276  001006                          BNE     5$
21984  134300  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
21985  134302  104401                          TYPE                    ;;TYPE A CR AND LF
21986  134304  001175                          $CRLF
21987  134306  105037  134514                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
21988  134312  000755                          BR      2$              ;;GET NEXT CHARACTER
21989  134314  004737  134376          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
21990  134320  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
21991  134324  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
21992  134326  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
21993                                           ;;AND THE NULL CHAR.
21994  134332  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
21995  134336  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
21996  134340  004737  134376                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
21997  134344  105337  134514                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
21998  134350  000770                          BR      7$              ;;LOOP
21999
22000                                   ;HORIZONTAL TAB PROCESSOR
22001
22002  134352  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
22003  134356  004737  134376          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
22004  134362  132737  000007  134514          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
22005  134370  001372                          BNE     9$              ;;TAB STOP
22006  134372  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
22007  134374  000724                          BR      2$              ;;GET NEXT CHARACTER
22008  134376                          $TYPEC:
22009  134376  105777  044542                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
22010  134402  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
22011  134404  017746  044536                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
22012  134410  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
22013  134414  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
22014  134420  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
```

```
 22015  134422                          101$:                                                           ;MJD001
 22016  134422  105777  044516                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
 22017  134426  100375                          BPL     101$                                            ;MJD001
 22018  134430  117716  044512                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
 22019  134434  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
 22020  134440  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
 22021  134444  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
 22022  134446                          102$:                                                           ;MJD001
 22023  134446  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
 22024  134450                          10$:                                                            ;MJD001
 22025  134450  105777  044474                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 22026  134454  100375                          BPL     10$                                             ;MJD001
 22027  134456  116677  000002  044466          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 22028  134464  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 22029  134472  001003                          BNE     1$              ;;BRANCH IF NO
 22030  134474  105037  134514                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 22031  134500  000406                          BR      $TYPEX          ;;EXIT
 22032  134502  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 22033  134510  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 22034  134512  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 22035  134514  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 22036  134516  000207          $TYPEX: RTS     PC
 22037
 22038                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 22039
 22040                          ;;*******  ************************************************************
 22041                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 22042                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 22043                          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 22044                          ;*CALL:
 22045                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 22046                          ;*      TYPOS                   ;;CALL FOR TYPEOUT
 22047                          ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 22048                          ;*      .BYTE   M               ;;M=1 OR 0
 22049                          ;*                                      ;;1=TYPE LEADING ZEROS
 22050                          ;*                                      ;;0=SUPPRESS LEADING ZEROS
 22051                          ;*
 22052                          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 22053                          ;*$TYPOS OR $TYPOC
 22054                          ;*CALL:
 22055                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 22056                          ;       TYPON                   ;;CALL FOR TYPEOUT
 22057                          ;
 22058                          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 22059                          ;*CALL:
 22060                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 22061                          ;*      TYPOC                   ;;CALL FOR TYPEOUT
 22062
 22063  134520  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 22064  134524  116637  000001  134743          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 22065  134532  112637  134745                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 22066  134536  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 22067  134542  000406                          BR      $TYPON
 22068  134544  112737  000001  134743  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 22069  134552  112737  000006  134745          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 22070  134560  112737  000005  134742  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
```

```
22071  134566  010346                        MOV     R3,-(SP)        ;;SAVE R3
22072  134570  010446                        MOV     R4,-(SP)        ;;SAVE R4
22073  134572  010546                        MOV     R5,-(SP)        ;;SAVE R5
22074  134574  113704  134745                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
22075  134600  005404                        NEG     R4
22076  134602  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
22077  134606  110437  134744                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
22078  134612  113704  134743                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
22079  134616  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
22080  134622  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
22081  134624  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
22082  134626  000404                        BR      3$              ;;GO DO MSB
22083  134630  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
22084  134632  006105                        ROL     R5
22085  134634  006105                        ROL     R5
22086  134636  010503                        MOV     R5,R3
22087  134640  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
22088  134642  105337  134744                DECB    $OMODE          ;;TYPE THIS DIGIT?
22089  134646  100016                        BPL     7$              ;;BR IF NO
22090  134650  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
22091  134654  001002                        BNE     4$              ;;TEST FOR 0
22092  134656  005704                        TST     R4              ;;SUPPRESS THIS 0?
22093  134660  001403                        BEQ     5$              ;;BR IF YES
22094  134662  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
22095  134664  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
22096  134670  052703  000040          5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
22097  134674  110337  134740                MOVB    R3,8$           ;;SAVE FOR TYPING
22098  134700  104401  134740                TYPE    ,8$             ;;GO TYPE THIS DIGIT
22099  134704  105337  134742          7$:   DECB    $OCNT           ;;COUNT BY 1
22100  134710  003347                        BGT     2$              ;;BR IF MORE TO DO
22101  134712  002402                        BLT     6$              ;;BR IF DONE
22102  134714  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
22103  134716  000744                        BR      2$              ;;GO DO THE LAST DIGIT
22104  134720  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
22105  134722  012604                        MOV     (SP)+,R4        ;;RESTORE R4
22106  134724  012603                        MOV     (SP)+,R3        ;;RESTORE R3
22107  134726  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
22108  134734  012616                        MOV     (SP)+,(SP)
22109  134736  000002                        RTI                     ;;RETURN
22110  134740     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
22111  134741     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
22112  134742     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
22113  134743     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
22114  134744  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
22115                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
22116
22117                                ;;*****************************************************************
22118                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
22119                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
22120                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
22121                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
22122                                ;*REPLACED WITH SPACES.
22123                                ;*CALL:
22124                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
22125                                ;*      TYPDS                   ;;GO TO THE ROUTINE
22126
```

```
 22127  134746                          $TYPDS:
 22128  134746  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 22129  134750  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 22130  134752  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 22131  134754  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 22132  134756  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 22133  134760  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 22134  134764  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 22135  134770  100004                          BPL     1$              ;;BR IF INPUT IS POS.
 22136  134772  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 22137  134774  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 22138  135002  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 22139  135004  012703  135162                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 22140  135010  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 22141  135014  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 22142  135016  016001  135152                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 22143  135022  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 22144  135024  002402                          BLT     4$              ;;BR IF DONE
 22145  135026  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 22146  135030  000774                          BR      3$
 22147  135032  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 22148  135034  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
 22149  135036  001002                          BNE     5$              ;;FALL THROUGH IF 0
 22150  135040  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 22151  135042  100407                          BMI     7$              ;;BR IF YES
 22152  135044  106316                  5$:     ASLB    (SP)            ;;MSD?
 22153  135046  103003                          BCC     6$              ;;BR IF NO
 22154  135050  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 22155  135056  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 22156  135062  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 22157  135066  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 22158  135070  005720                          TST     (R0)+           ;;JUST INCREMENTING
 22159  135072  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
 22160  135076  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
 22161  135100  003002                          BGT     8$              ;;GO TO EXIT
 22162  135102  010502                          MOV     R5,R2           ;;GET THE LSD
 22163  135104  000764                          BR      6$              ;;GO CHANGE TO ASCII
 22164  135106  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 22165  135110  100003                          BPL     9$              ;;BR IF NO
 22166  135112  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 22167  135120  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 22168  135122  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 22169  135124  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 22170  135126  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 22171  135130  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 22172  135132  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 22173  135134  104401  135162                  TYPE    ,#DBLK          ;;NOW TYPE THE NUMBER
 22174  135140  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 22175  135146  012616                          MOV     (SP)+,(SP)
 22176  135150  000002                          RTI                     ;;RETURN TO USER
 22177  135152  023420                  $DTBL:  10000.
 22178  135154  001750                          1000.
 22179  135156  000144                          100.
 22180  135160  000012                          10.
 22181  135162  000004                  $DBLK:  .BLKW   4
 22182                                  .SBTTL  TTY INPUT ROUTINE
```

```
22183
22184                                   ;;*****************************************************************
22185                                   .ENABL  LSB
22186
22187                                   ;;*****************************************************************
22188                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
22189                                   ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
22190                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
22191                                   ;*WHEN OPERATING IN TTY FLAG MODE.
22192  135172  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
22193  135200  001074                         BNE     15$             ;;BRANCH IF NO
22194  135202  105777  043736                 TSTB    @$TKS           ;;CHAR THERE?
22195  135206  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
22196  135210  117746  043732                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
22197  135214  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
22198  135220  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
22199  135224  001062                         BNE     15$             ;;NO, RETURN TO USER
22200  135226  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
22201  135234  001456                         BEQ     15$             ;;BRANCH IF YES
22202
22203  135236  104401  135727                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
22204  135242  104401  135734  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
22205  135246  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
22206  135252  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
22207  135254  104401  135745                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
22208  135260  005046                 19$:    CLR     -(SP)           ;;CLEAR COUNTER
22209  135262  005046                         CLR     -(SP)           ;;THE NEW SWR
22210  135264  105777  043654          7$:    TSTB    @$TKS           ;;CHAR THERE?
22211  135270  100375                         BPL     7$              ;;IF NOT TRY AGAIN
22212
22213  135272  117746  043650                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
22214  135276  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
22215
22216
22217
22218  135302  021627  000025          9$:    CMP     (SP),#25        ;;IS IT A CONTROL-U?
22219  135306  001005                         BNE     10$             ;;BRANCH IF NOT
22220  135310  104401  135722                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
22221  135314  062706  000006          20$:   ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
22222  135320  000757                         BR      19$             ;;LET'S TRY IT AGAIN
22223
22224
22225  135322  021627  000015          10$:   CMP     (SP),#15        ;;IS IT A <CR>?
22226  135326  001022                         BNE     16$             ;;BRANCH IF NO
22227  135330  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
22228  135334  001403                         BEQ     11$             ;;BRANCH IF YES
22229  135336  016677  000002  043574         MOV     2(SP),@SWR      ;;SAVE NEW SWR
22230  135344  062706  000006          11$:   ADD     #6,SP           ;;CLEAR UP STACK
22231  135350  104401  001175          14$:   TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
22232  135354  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
22233  135362  001003                         BNE     15$             ;;BRANCH IF NOT
22234  135364  012777  000100  043552         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
22235  135372  000002                 15$:    RTI                     ;;RETURN
22236  135374  004737  134376          16$:   JSR     PC,$TYPEC       ;;ECHO CHAR
22237  135400  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
22238  135404  002420                         BLT     18$             ;;BRANCH IF YES
```

```
 22239  135406  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 22240  135412  003015                          BGT     18$             ;;BRANCH IF YES
 22241  135414  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 22242  135420  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 22243  135424  001403                          BEQ     17$             ;;BRANCH IF YES
 22244  135426  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 22245  135430  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
 22246  135432  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
 22247  135434  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 22248  135440  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 22249  135444  000707                          BR      7$              ;;GET THE NEXT ONE
 22250  135446  104401  001174          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 22251  135452  000720                          BR      20$             ;;SIMULATE CONTROL-U
 22252                                  .DSABL  LSB
 22253
 22254
 22255                                  ;;*****************************************************************
 22256                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 22257                                  ;*CALL:
 22258                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 22259                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 22260                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 22261                                  ;
 22262
 22263  135454  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 22264  135456  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 22265  135464  105777  043454          1$:     TSTB    @#TKS           ;;WAIT FOR
 22266  135470  100375                          BPL     1$              ;;A CHARACTER
 22267  135472  117766  043450  000004          MOVB    @#TKB,4(SP)     ;;READ THE TTY
 22268  135500  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 22269  135506  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 22270  135514  001013                          BNE     3$              ;;BRANCH IF NO
 22271  135516  105777  043422          2$:     TSTB    @#TKS           ;;WAIT FOR A CHARACTER
 22272  135522  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
 22273  135524  117746  043416                  MOVB    @#TKB,-(SP)     ;;GET CHARACTER
 22274  135530  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 22275  135534  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 22276  135540  001366                          BNE     2$              ;;IF NOT DISCARD IT
 22277  135542  000750                          BR      1$              ;;YES, RESUME
 22278  135544  026627  000004  000021  3$:     CMP     4(SP),#$XON     ;;IS IT A RANDOM XON?                   ;RAN001
 22279  135552  001744                          BEQ     1$              ;;BRANCH IF YES                         ;RAN001
 22280  135554  026627  000004  000140          CMP     4(SP),#140      ;;IS IT UPPER CASE?
 22281  135562  002407                          BLT     4$              ;;BRANCH IF YES
 22282  135564  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 22283  135572  003003                          BGT     4$              ;;BRANCH IF YES
 22284  135574  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 22285  135602  000002                  4$:     RTI                     ;;GO BACK TO USER
 22286                                  ;;*****************************************************************
 22287                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 22288                                  ;*CALL:
 22289                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 22290                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 22291                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 22292
 22293  135604  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 22294  135606  012703  135712          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
```

E2


```
22295  135612  022703  135722          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
22296  135616  101405                          BLOS    4$              ;;BR IF YES
22297  135620  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
22298  135622  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
22299  135624  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
22300  135630  001003                          BNE     3$              ;;SKIP IF NOT
22301  135632  104401  001174          4$:     TYPE    ,$QUES          ;;TYPE A '?'
22302  135636  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
22303  135640  111337  135710          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
22304  135644  104401  135710                  TYPE    ,9$
22305  135650  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
22306  135654  001356                          BNE     2$              ;;LOOP IF NOT RETURN
22307  135656  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
22308  135662  104401  001176                  TYPE    ,$LF            ;;TYPE A LINE FEED
22309  135666  012603                          MOV     (SP)+,R3        ;;RESTORE R3
22310  135670  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
22311  135672  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
22312  135700  012766  135712  000004          MOV     #$TTYIN,4(SP)
22313  135706  000002                          RTI                     ;;RETURN
22314  135710     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
22315  135711     000                  .BYTE   0                       ;;TERMINATOR
22316  135712  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
22317  135722  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
22318  135727     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
22319  135734  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
22320  135742  020075     000
22321  135745     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
22322  135752  036440  000040
22323                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
22324
22325                                  ;;**************************************************************
22326                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
22327                                  ;*CHANGE IT TO BINARY.
22328                                  ;*CALL:
22329                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
22330                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
22331                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
22332
22333  135756  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
22334  135760  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
22335  135766  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
22336  135770  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
22337  135772  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
22338  135774  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
22339  135776  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
22340  136000  005001                          CLR     R1              ;;CLEAR DATA WORD
22341  136002  005002                          CLR     R2
22342  136004  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
22343  136006  001412                          BEQ     3$              ;;IF ZERO GET OUT
22344  136010  006301                          ASL     R1              ;;*2
22345  136012  006102                          ROL     R2
22346  136014  006301                          ASL     R1              ;;*4
22347  136016  006102                          ROL     R2
22348  136020  006301                          ASL     R1              ;;*8
22349  136022  006102                          ROL     R2
22350  136024  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
```

F2


```
 22351  136030  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
 22352  136032  000764                          BR      2$              ;;LOOP
 22353  136034  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
 22354  136036  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
 22355  136042  010237  136056                  MOV     R2,$HIOCT
 22356  136046  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 22357  136050  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 22358  136052  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 22359  136054  000002                          RTI                     ;;RETURN
 22360  136056  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 22361                                  .SBTTL  TRAP DECODER
 22362
 22363                                  ;;**************************************************************
 22364                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 22365                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 22366                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 22367                                  ;*GO TO THAT ROUTINE.
 22368
 22369  136060  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 22370  136062  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
 22371  136066  005740                          TST     -(R0)           ;;BACKUP BY 2
 22372  136070  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 22373  136072  006300                          ASL     R0              ;;POSITION FOR INDEXING
 22374  136074  016000  136114                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 22375  136100  000200                          RTS     R0              ;;GO TO ROUTINE
 22376
 22377
 22378                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
 22379
 22380  136102  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
 22381  136104  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
 22382  136112  000002                          RTI                     ;;RESTORE THE PSW
 22383
 22384                                  .SBTTL  TRAP TABLE
 22385
 22386                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 22387                                  ;*BY THE "TRAP" INSTRUCTION.
 22388
 22389                                  ;       ROUTINE
 22390                                  ;       -------
 22391  136114  136102                  $TRPAD: .WORD   $TRAP2
 22392  136116  134164                          $TYPE   ;;CALL-TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
 22393  136120  134544                          $TYPOC  ;;CALL-TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 22394  136122  134520                          $TYPOS  ;;CALL-TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 22395  136124  134560                          $TYPON  ;;CALL-TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 22396  136126  134746                          $TYPDS  ;;CALL-TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
 22397
 22398  136130  135242                          $GTSWR  ;;CALL-GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
 22399
 22400  136132  135172                          $CKSWR  ;;CALL-CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
 22401  136134  135454                          $RDCHR  ;;CALL-RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 22402  136136  135604                          $RDLIN  ;;CALL-RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 22403  136140  135756                          $RDOCT  ;;CALL-RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
 22404                                  .SBTTL  POWER DOWN AND UP ROUTINES
 22405
 22406                                  ;;**************************************************************
```

```
22407                                          ;POWER DOWN ROUTINE
22408  136142  012737  136302  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
22409  136150  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
22410  136156  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
22411  136160  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
22412  136162  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
22413  136164  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
22414  136166  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
22415  136170  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
22416  136172  017746  042742                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
22417  136176  010637  136306                  MOV     SP,$SAVR6       ;;SAVE SP
22418  136202  012737  136214  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
22419  136210  000000                          HALT
22420  136212  000776                          BR      .-2             ;;HANG UP
22421
22422                                          ;;**************************************************************
22423                                          ;POWER UP ROUTINE
22424  136214  012737  136302  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
22425  136222  013706  136306                  MOV     $SAVR6,SP       ;;GET SP
22426  136226  005037  136306                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
22427  136232  005237  136306          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
22428  136236  001375                          BNE     1$              ;;OF  WORD
22429  136240  012677  042674                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
22430  136244  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
22431  136246  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
22432  136250  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
22433  136252  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
22434  136254  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
22435  136256  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
22436  136260  012737  136142  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
22437  136266  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
22438  136274  104401                          TYPE                    ;REPORT THE POWER FAILURE
22439  136276  136310                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
22440  136300  000002                          RTI
22441  136302  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
22442  136304  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
22443  136306  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
22444  136310  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
22445  136316  000122
22446                                          .EVEN
22447          000001                          .END
```

```
ABASE = 000000            623     664
ABOEXT  105542          17467   17476   17478#
ABORT   133556          21805   21852#
ABORTC  133610          21857   21865#
ABORTE  133654          21861   21875#
ABORTI  104270          17073#  17094
ABORTR  104324          17060   17066   17083#
ABORTZ  133700          21855   21874   21879#
ABORTO  043610           9176    9292#
ABORT7  043776           9187    9354#
ABROUT  105420          17320   17412#
ACCC  = ****** U         4289
ACDW1 = 000000            623     666
ACDW2 = 000000            623     667
ACPUOP= 000000            623     638
ACTCHS  002704           1179#  18524*  18529*  18539   18553*  18558*  18573   18610*
ADCC  = ****** U         4260
ADDLSB  110156          17970*  18045   18050   18060*  18061   18074#  18225*  18287   18292   18304*  18305
ADDT    051772          10482   10488   10494   10500   10507   10521#
ADDTRP  132512           7989    8011   21605#
ADDW0 = 000000            623     668
ADDW1 = 000000            623     669
ADDW10= 000000            623     678
ADDW11= 000000            623     679
ADDW12= 000000            623     680
ADDW13= 000000            623     681
ADDW14= 000000            623     682
ADDW15= 000000            623     683
ADDW2 = 000000            623     670
ADDW3 = 000000            623     671
ADDW4 = 000000            623     672
ADDW5 = 000000            623     673
ADDW6 = 000000            623     674
ADDW7 = 000000            623     675
ADDW8 = 000000            623     676
ADDW9 = 000000            623     677
ADEVCT= 000000            623     629
ADEVM = 000000            623     665
AENV  = 000000            623     634
AENVM = 000000            623     635
AFATAL= 000000            623     626
ALCC  = ****** U         4519
ALLCTR  003112          12520   17735*  17749*  17759*  17782*
ALROTS  021322          52710
ALR1TS  021400          52950
ALR2TS  021456          53190
ALR3TS  021534          53430
ALR4TS  021612          53670
ALR5TS  021670          53910
ALTREG= ****** U         5180
ALTR1 = ****** U         5295
ALTR2 = ****** U         5319
ALTR3 = ****** U         5343
ALTR4 = ****** U         5367
ALTR5 = ****** U         5391
AMADR1= 000000            623     651
```

```
AMADR2= 000000            623     655
AMADR3= 000000            623     658
AMADR4= 000000            623     661
AMAMS1= 000000            623     645
AMAMS2= 000000            623     653
AMAMS3= 000000            623     656
AMAMS4= 000000            623     659
AMSGAD= 000000            623     631
AMSGLG= 000000            623     632
AMSGTY= 000000            623     625
AMTYP1= 000000            623     646
AMTYP2= 000000            623     654
AMTYP3= 000000            623     657
AMTYP4= 000000            623     660
APASS = 000000            623     628
APRIOR= 000000            623
APTCSU= 000040          21941#  21972
APTENV= 000001           5666   18461   18979   19296   19340   19682   19773   21823   21897   21939#  21965
APTSIZ= 000200           1475   21938#
APTSPO= 000100          21899   21940#  21967
ARCC  = ****** U         4551
ASWREG= 000000            623     636
ATESTN= 000000            623     627
AUNIT = 000000            623     630
AUSWR = 000000            623     637
AVECT1= 000000            623     662
AVECT2= 000000            623     663
A11   = ****** U         2908
A126    025040           6137#   6145
A55   = ****** U         3169
BA      002650           1162#  20100#  20134#  20475#  21526#  21537#
BACDAT  102536          16454   16475#
BA2     002670           1172#  20476#
BB11  = ****** U         2944
BB22  = ****** U         3053
BCCC  = ****** U         4143
BCR   = 177524           1190#  19161
BCSR  = 177520           1192#  17821#  17823#  17967#  18066#  18211#  18311#  18420   18421#  18428#  18430   18433#  18435
                        18442#  18443   18450#  18451   18456#  18457   18465#  18467   18470#  18471#  18517   18518#  18580
                        18601   18602#  18603#  18627#  18663   18746   18830   18835   18836#  18837   18841#  18842   18857#
                        18901   18906   18907#  18908   18912#  18913   18922#  18924#  18939   18943   18949#  18985   18990
                        18991#  19049   19052   19057   19062#  19064#  19067#  19097#  19551   19552#  19582#  19942   19946#
                        19951#  19961#  20498   20500#  20520#  20526#  20545#  21220
BC00  = ****** U         2831
BC11  = ****** U         2974
BC22  = ****** U         3082
BDR   = 177524           1191#  19693#  19707#
BFA     052024          10478   10484   10490   10496   10503   10510   10527#
BFAC1   051612          10479   10482#
BFAC2   051632          10485   10488#
BFAC3   051652          10491   10494#
BFAC4   051672          10497   10500#
BFAC5   051714          10504   10507#
BFAE    052064          10511   10514   10537#
BFB     052062          10528   10530   10532   10534#
BGNTLP  107474          17988#  18054
```

J2


```
BIT0   = 000001          510#
BIT00  = 000001          500#   510   10015  16695  17002  17202  17218  17370  17380  19300  19305  19553  19554
                        19562  19573  19574  19893  19907  19947  20126  20499  21509  21511  21523  21528  21542
                        21547
BIT01  = 000002          499#   509   16304  16342  17616  17622  17641  17647
BIT02  = 000004          498#   508   16294  16389  16394  16399  16588  16594  16604  16676  16686  16692  16756
                        16770  16817  16822  16827  16832  16921  16937  17005  17751  17763  17776  18250  18256
                        18263  19242  19243  19279  19300  19305  19395  19400  19416  19458  19474  19488  19550
                        19570  19576  19591  19593  19626  19641  19658  19660  19799  19893  19899  20057  20075
                        20089  20111  20142  20181
BIT03  = 000010          497#   507   6853   16335  18258  18443
BIT04  = 000020          496#   506   7624   9744   9758   17631  17656  17729  17794  18173  18450  18451  18453
                        18456  18457  18465  18467  18470  19787  19816  20009
BIT05  = 000040          495#   505   1486   10022  16316  18518  18603  18661  18744  18828  18899  18941  18983
                        19047  19065  19077  19159  19543  19678  19769  19979  20039  20217  20299  20353  20360
                        20391  20463  21428  21691
BIT06  = 000100          494#   504   7652   7675   16917  16932  16934  17147  17202  17218  18168  18518  18602
                        18754  18755  18766  18776  18792  18945  18948  18994  19006  19069  19084  19348  19349
                        19352  19353  19357  19360  19368  19382  19391  19410  19471  19482  19484  19501  19689
                        21422
BIT07  = 000200          493#   503   16583  17071  17327  17344  17355  17924  18518  18767  18773  19003  19161
                        19999  21425
BIT08  = 000400          492#   502   9707   17227  17821  17823  17967  18066  18211  18311  19946  19951  21751
BIT09  = 001000          491#   501   16814  16835  17204  19552  19882  19913  20084  21441  21523  21542  21761
                        21834
BIT1   = 000002          509#
BIT10  = 002000          490#   17000  17069  17202  17218  19947  19950  19996  19998  21811
BIT11  = 004000          489#   5191   5202   5203   5231   5238   5245   5413   19568  19589  21768
BIT12  = 010000          488#   18663  18746  18830  18836  18837  18841  18842  18844  18922  18924  18939  18943
                        18985  18991  19049  20104  20173
BIT13  = 020000          487#   18901  18907  18908  18912  18913  18915  19568  19589  21818
BIT14  = 040000          486#   19656  21737
BIT15  = 100000          485#   16669  16673  16679  16682  17628  17653  17738  17740  18212  19568  19589  19656
                        19909  20125
BIT2   = 000004          508#   17530  17541  17766  20070
BIT3   = 000010          507#   17527  17538  17550
BIT4   = 000020          506#   17547
BIT5   = 000040          505#   1416
BIT6   = 000100          504#   1419   19643  21280
BIT7   = 000200          503#   19643  21280
BIT8   = 000400          502#   21280
BIT9   = 001000          501#
BIOO   = ****** U        2801
BNO      043540          9175   9272#
BN1      043730          9186   9197   9335#
BPTO   = ****** U        6995
BPTOA    030452          7001   7007#
BPTOB    030460          7003   7010#
BPTVEC= 000014           517#
BSCC   = ****** U        4170
BT     = ****** U        7449
BTCC   = ****** U        4117
BTER     051154          10353  10355  10357  10360#
BTEXP    003040          12444  10316  10332  10335  10338  10341  10344  10348  10352  10354  10356  10358  10457
                        10459* 10460* 10461* 10462* 10463  10466  10469  10471  10473  10475  10516* 10517* 10518*
                        10519* 10521* 10522* 10523* 10524* 10527  10529  10531  10533
```

K2


```
BTGO    050754          10321# 10421
BTO   = ****** U         7471
BTRES   003050           1245# 10317  10333* 10336* 10339* 10342* 10352  10354  10356  10358  10369* 10376* 10383*
                        10390* 10398* 10406* 10429  10431  10433  10435  10458  10477* 10483* 10489* 10495* 10502*
                        10509* 10527  10529  10531  10533
BTTST   051420          10370  10377  10384  10391  10399  10407  10429#
BTTSTE  051446          10430  10432  10434  10436#
BT1   = ****** U        10305
BYPARR  103012          16581  16615#
B66   = ****** U         3192
CBIT  = ****** U         4491
CB44  = ****** U         3141
CCR   = 177746           1193# 16156  16159  16205* 16206* 16207  16209  16294* 16296* 16335* 16350* 16389* 16391*
                        16583* 16610* 16753* 16767* 16814* 16835* 16915* 16917* 16932* 16934* 16944* 16949* 17000*
                        17002* 17069* 17071* 17147* 17149* 17202* 17204* 17216* 17218* 17220* 17225* 17227* 17325*
                        17327* 17342* 17344* 17353* 17355* 17368* 17370* 17378* 17380* 17389* 17469* 17471* 17478*
                        17616* 17622* 17641* 17647* 17733* 17751* 17763* 17776* 17791* 17812* 17930* 17962* 18174*
                        18205* 19778* 19882* 19913* 19947* 19950* 19960* 19996* 19998* 19999* 20013* 20070* 20072*
                        20084* 20086* 21194
CCRTBL  105360          17322  17338  17365  17393#
CHECK   015132           4014   4035#
CHECK1  015210           4036   4055#
CHECK2  015266           4056   4075#
CHECK7  053174          10789  10793  10801  10805  10811#
CHEC10  053414          10851  10855  10863  10867  10873#
CHEC26  056176          11490  11499  11508  11519  11528  11539#
CHEC27  056504          11596  11610  11624  11633#
CHEC30  057014          11690  11704  11718  11727#
CHEC32  057406          11823  11832  11841  11850#
CHEC33  057662          11903  11912  11921  11930#
CHEK7   053200          10812# 10816
CHK10   053426          10875  10878#
CHK7    053206          10813  10816#
CH10    053420          10874# 10878
CKSWR = 104407          21736  21807  21833  22400#
CLRD  = ****** U        14763
CLRI  = ****** U        14795
CMPD  = ****** U        13046
CMPRTN  064204          13059  13068  13077  13085  13107#
COUNT   003060           1246#  6796*  6802*  6803*  6809*  6815*  6816*  12026  12069  12110  12147  12178  12213
                        12250  12281  12296  12309  12323  12358  12373  12405  12420  12448  12460  12493  12503
                        12517  12528  12561  12589  12619  12640  12658  12676  12709  12724  12754  12766  12798
                        12828  12849  12879  12900  12930  12963  12987  13019  13039  13132  13366  13558  13770
                        13971  14183  14192  14474  14483  14587  14719  14783  15127  15306  15434  15727  15949
                        21665* 21670* 21671*
CPEREG= 177766           1206#  6839*  6853   6861*  6873*  6883*  6892*  6899*  6908*  6922*  6931*  6944*  6954*
                         6969*  6975*  6990*  6998*  7011*  7021*  7037*  7615*  7624   7633*  7642*  7652   7661*
                         7675   7687*  7697*  7710   7713*  7725*  7985*  8003*  9806*  10016* 10022  10032*
CPUTST= ****** U         1512
CR    = 000015           425#  22028  22038
CRLF  = 000200           426#   1508  21982  22038
CSR1    002644          1160#  19982  20042  20102* 20136* 20219  20224  20301  20309  20355  20414  20481* 20484
                        20491  20493  21465  21490  21525* 21540* 21544*
CSR12   002664          1170#  20393  20411  20466  20482* 20486  20490  20494  21469
CSR2    002646          1161#  20104  20173  20239* 20268* 20316* 20360* 20362* 20365* 20413* 20473* 21523* 21528*
                        21529  21542* 21547* 21548
```

L2


```
CSR22    002666          1171#  20417*  20474*
CTSCC  = ****** U        4227
CURADD   110172         17734*  17739   17741*  17743*  17744*  17754*  17755*  17761   17764   17768   17772   17775*  17780*
                        17986*  17993   18007   18014*  18016*  18017   18021   18041*  18045*  18046   18048*  18050*  18051
                        18053*  18065   18080#  18237*  18241   18248   18254*  18261   18266   18283*  18287*  18288   18290*
                        18292*  18293   18295*  18309
CURDAT   110164         17976*  18006   18008   18027*  18031*  18077#  18226*  18247   18268*
C121A    023054          5675    5678#
C121B    023074          5680    5683#
C121C    023114          5685    5688#
DAPABO   104062         16909   16954#
DATA     002654          1164#  20159   20176   20479*  21520*
DATA2    002674          1174#  20480*
DATBO    117722         20090   20095#
DATI     120222         20127   20153#
DATVER   132672         12025   12068   12109   12146   12177   12212   12249   12280   12295   12308   12322   12357   12372
                        12404   12419   12447   12459   12492   12502   12516   12527   12560   12588   12618   12639   12657
                        12675   12708   12723   12753   12765   12797   12827   12848   12878   12899   12929   12962   12986
                        13018   13038   13131   13557   13970   14473   14482   14586   14782   15126   15433   15726   21668#
DATVFR   132654         13365   13769   14182   14191   14718   15305   15948   21663#
DAT1     132704         21666   21671#  21674
DAO    = ****** U        2737
DCCC   = ****** U        4195
DCOUNT   110146          7878*   7896*  17760*  17777*  17987*  17988   18012   18025   18035   18040*  18070#  18236*  18269
                        18274   18282*  18296
DDISP  = 177570           432#    602    1463
DH1      127564           708     713     718     783     788     793     808     813     828     833     838     843     853
                          858     863     873     883     888     918     923     928     933     938     943     948     953
                          958     963     973     978     983     988     998    1003    1013    1018    1023    1028    1033
                         1038    1043    1058    1063    1083    1093    1098    1103    1108    1113    1118    1123    1128
                         1133    1138    1143    1148    1153   21086#
DH105    130525          1048    1053    1073    1078   21177#
DH115    130557          1088   21183#
DH24     130050           803   21120#
DH27     130114           818     823   21127#
DH4      127611           723     763     778     848     908     913    1008    1068   21091#
DH41     130157           868   21134#
DH43     130230           878   21142#
DH47     130330           893     898     903   21153#
DH5      127676           728     733     768     773     798   21101#
DH65     130415           968   21163#
DH7      127775           738     743     748     753     758   21112#
DH72     130461           993   21170#
DISPLA   001142           602#   1463*   1471*  21783*  21810*
DISPRE   000174           533#   1471
DIVD   = ****** U       13382
DIVF   = ****** U       13141
DMAPAR   117204         19937   19946#
DMARD    132244         20158   20172   21537#
DMATRN   132162         20001   20054   20071   20085   20103   20137   21520#
DM0    = ****** U        2721
DM27   = ****** U        2779
DPAREN   117244         19939   19955#
DSWR   = 177570           431#    601    1462
DS0    = ****** U        2757
DT1      130634           709     714     719     784     789     794     809     814     829     834     839     844     854
```

M2


```
                        859     864     874     884     889     919     924     929     934     939     944     949     954
                        959     964     974     979     984     989     999    1004    1014    1019    1024    1029    1034
                       1039    1044    1059    1064    1084    1094    1099    1104    1109    1114    1119    1124    1129
                       1134    1139    1149    1154   21193#
DT105   131112         1049    1054    1074    1079   21228#
DT115   131122         1089   21230#
DT130   131136         1144   21232#
DT14    130702          764   21200#
DT17    130714          779   21202#
DT24    130726          804   21204#
DT27    130736          819     824   21206#
DT35    130750          849   21208#
DT4     130642          724   21194#
DT41    130762          869     994   21210#
DT43    130772          879   21212#
DT47    131006          894     899   21214#
DT5     130654          729     734     769     774     799   21196#
DT50    131020          904   21216#
DT51    131032          909   21218#
DT52    131044          914   21220#
DT64    131056        21222#
DT65    131070          969   21224#
DT7     130670          739     744     749     754     759   21198#
DT75    131100         1009    1069   21226#
DVDSUB  066204        13395   13407   13419   13430   13441   13452   13463   13474   13486   13512#
DVFSUB  065242        13154   13164   13173   13181   13190   13198   13206   13214   13222   13230   13238   13246   13254
                      13263   13271   13280   13289   13298   13320#
D1      052404        10635#  10649   10654
D2      052416        10638#  10639#
D3      052420        10640#  10662
D4      052432        10646#  10679   10682
D5      052446        10647   10650#
D6      052456        10651   10653#
D7      052462        10652   10655#
EMTO  = ******  U      6951
EMTOA   030256         6957    6963#
EMTOB   030264         6959    6966#
EMTSAV  003764         1414#
EMTVEC= 000030          520#    1446#   1447#
EM1     122020          707   20553#
EM10    122272          742   20584#
EM100   125606         1022   20905#
EM101   125644         1027   20910#
EM102   125677         1032   20915#
EM103   125761         1037   20924#
EM104   126034         1042   20932#
EM105   126104         1047   20939#
EM106   126157         1052   20947#
EM107   126235         1057   20955#
EM11    122313          747   20587#
EM110   126272         1062   20960#
EM111   126334         1067   20966#
EM112   126412         1072   20974#
EM113   126475         1077   20983#
EM114   126561         1082   20992#
EM115   126602         1087   20995#
```

EM116	126652	1092	21002#
EM117	126733	1097	21011#
EM12	122335	752	20591#
EM120	126776	1102	21017#
EM121	127044	1107	21024#
EM122	127142	1112	21035#
EM123	127170	1117	21039#
EM124	127222	1122	21044#
EM125	127314	1127	21054#
EM126	127353	1132	21060#
EM127	127410	1137	21065#
EM13	122363	757	20595#
EM130	127452	1142	21071#
EM131	127477	1147	21075#
EM132	127527	1152	21080#
EM14	122412	762	20599#
EM15	122435	767	20603#
EM16	122471	772	20608#
EM17	122536	777	20615#
EM2	122054	712	20558#
EM20	122602	782	20621#
EM21	122635	787	20626#
EM22	122700	792	20632#
EM23	122737	797	20638#
EM24	123013	802	20646#
EM25	123056	807	20652#
EM26	123116	812	20658#
EM27	123163	817	20665#
EM3	122066	717	20560#
EM30	123225	822	20671#
EM31	123277	827	20679#
EM32	123324	832	20683#
EM33	123365	837	20689#
EM34	123427	842	20695#
EM35	123467	847	20701#
EM36	123515	852	20705#
EM37	123561	857	20712#
EM4	122100	722	20562#
EM40	123623	862	20718#
EM41	123670	867	20725#
EM42	123754	872	20734#
EM43	124002	877	20738#
EM44	124033	882	20743#
EM45	124073	887	20749#
EM46	124155	892	20758#
EM47	124245	897	20768#
EM5	122140	727	20568#
EM50	124332	902	20777#
EM51	124355	907	20781#
EM52	124407	912	20786#
EM53	124445	917	20792#
EM54	124501	922	20797#
EM55	124537	927	20803#
EM56	124573	932	20808#
EM57	124617	937	20812#
EM6	122173	732	20573#

```
EM60     124651            942   20817#
EM61     124717            947   20824#
EM62     124746            952   20828#
EM63     125017            957   20835#
EM64     125050            962   20840#
EM65     125107            967   20846#
EM66     125157            972   20853#
EM67     125216            977   20859#
EM7      122236            737   20579#
EM70     125253            982   20864#
EM71     125304            987   20869#
EM72     125340            992   20874#
EM73     125376            997   20879#
EM74     125422           1002   20883#
EM75     125453           1007   20888#
EM76     125504           1012   20893#
EM77     125554           1017   20900#
ENDHRT   102534          16452   16472#
ENDLUP   110112          18061#
ENDMOV   110174          17926   17945   17964   18082#
ENDTAG   111262          18170   18190   18208   18313#
ENDTLP   110062          17989   18055#
ERR      132652          21642#
ERRFP    132650          21641#
ERROR -  104000           4199    1534    1540    1550    1554    1568    1579    1589    1598    1608    1613    1618    1623
                          1632    1637    1642    1647    1656    1661    1666    1671    1680    1685    1690    1695    1704
                          1709    1714    1719    1728    1733    1738    1743    1752    1757    1762    1767    1777    1782
                          1787    1792    1804    1809    1821    1827    1839    1846    1860    1869    1883    1894    1908
                          1918    1933    1941    1957    1967    1981    1991    2012    2021    2032    2041    2058    2068
                          2087    2096    2100    2114    2127    2148    2156    2160    2168    2171    2193    2201    2205
                          2229    2233    2237    2244    2248    2269    2273    2278    2295    2303    2317    2333    2343
                          2360    2365    2382    2391    2410    2419    2439    2446    2465    2474    2495    2509    2530
                          2536    2560    2565    2584    2589    2608    2612    2628    2632    2652    2658    2676    2682
                          2704    2710    2731    2748    2752    2768    2774    2786    2791    2795    2814    2818    2823
                          2840    2846    2853    2860    2882    2890    2904    2921    2938    2954    2958    2964    2969
                          2985    2992    2998    3019    3025    3042    3048    3067    3074    3078    3094    3099    3105
                          3112    3130    3134    3152    3160    3166    3182    3187    3211    3216    3234    3237    3254
                          3264    3281    3292    3308    3320    3338    3345    3366    3386    3404    3422    3438    3452
                          3467    3482    3485    3493    3500    3504    3512    3516    3525    3529    3539    3548    3655
                          3675    3684    3690    3702    3706    3729    3737    3749    3757    3768    3776    3789    3797
                          3809    3817    3829    3837    3848    3856    3880    3896    3913    3928    3942    3946    3955
                          3959    3970    3975    3980    3985    3992    3997    4002    4007    4019    4024    4029    4040
                          4045    4050    4060    4065    4070    4078    4083    4088    4104    4112    4129    4138    4155
                          4165    4181    4190    4206    4222    4239    4248    4255    4271    4278    4284    4299    4309
                          4316    4331    4340    4350    4358    4363    4379    4388    4406    4421    4435    4441    4448
                          4454    4468    4474    4480    4486    4503    4506    4513    4516    4528    4534    4541    4547
                          4560    4567    4573    4581    4597    4608    4624    4633    4642    4665    4678    4689    4696
                          4712    4722    4742    4745    4749    4764    4770    4773    4778    4791    4797    4802    4806
                          4815    4827    4836    4846    4856    4866    4876    4886    4896    4906    4917    4923    4929
                          4935    4941    4947    4953    4959    4965    4971    4981    4987    4993    4999    5005    5011
                          5017    5023    5029    5035    5043    5046    5050    5054    5058    5062    5067    5070    5074
                          5078    5083    5086    5090    5094    5099    5102    5106    5110    5131    5141    5149    5154
                          5158    5162    5166    5170    5174    5193    5205    5209    5213    5217    5221    5225    5229
                          5248    5252    5256    5260    5264    5268    5276    5281    5286    5291    5300    5305    5310
                          5315    5324    5329    5334    5339    5348    5353    5358    5363    5372    5377    5382    5387
                          5396    5401    5406    5411    5427    5431    5445    5449    5463    5467    5471    5536    5545
```

5549	5558	5562	5566	5602	5606	5613	5617	5623	5639	5646	5650	5654
5658	5676	5681	5686	5692	5698	5702	5706	5725	5732	5737	5742	5761
5772	5848	5852	5858	5863	5871	5886	5919	5923	5929	5933	5941	5956
5976	5980	5984	5988	6008	6012	6017	6022	6026	6138	6143	6150	6161
6165	6169	6177	6181	6185	6195	6200	6204	6208	6216	6220	6224	6240
6244	6248	6252	6257	6398	6402	6410	6414	6418	6428	6432	6436	6441
6564	6568	6572	6576	6585	6589	6593	6597	6610	6614	6618	6622	6627
6800	6806	6812	6819	6825	6831	6850	6857	6880	6889	6897	6917	6941
6964	6985	7008	7032	7055	7081	7097	7099	7103	7107	7124	7128	7132
7136	7153	7155	7159	7168	7170	7174	7188	7192	7196	7213	7217	7225
7229	7244	7248	7252	7269	7273	7280	7284	7304	7309	7337	7341	7345
7359	7363	7370	7374	7397	7401	7426	7440	7456	7460	7464	7480	7484
7491	7495	7511	7515	7519	7534	7538	7545	7549	7565	7569	7573	7590
7594	7601	7605	7623	7626	7629	7632	7651	7654	7657	7660	7673	7677
7680	7683	7686	7700	7703	7706	7709	7712	7730	7749	7756	7798	7805
7821	7823	7860	7892	7913	7921	7960	7966	7998	8039	8043	8051	8062
8076	8089	8100	8111	8122	8133	8143	8152	8162	8174	8179	8188	8193
8199	8204	8212	8217	8225	8230	8238	8243	8249	8270	8276	8282	8288
8295	8301	8316	8323	8362	8366	8372	8376	8382	8386	8392	8396	8403
8407	8413	8417	8454	8458	8464	8468	8474	8478	8484	8488	8495	8499
8505	8509	8522	8529	8536	8543	8552	8556	8560	8564	8588	8594	8599
8603	8607	8612	8616	8625	8630	8634	8639	8643	8650	8655	8659	8663
8688	8694	8699	8703	8707	8712	8716	8725	8730	8734	8739	8743	8752
8757	8761	8766	8770	8777	8782	8786	8790	8816	8822	8827	8831	8835
8840	8849	8854	8858	8863	8872	8877	8881	8886	8893	8898	8902	8928
8934	8939	8943	8947	8952	8961	8966	8970	8975	8984	8989	8993	8998
9005	9010	9014	9086	9091	9095	9099	9110	9115	9119	9144	9149	9153
9157	9217	9222	9227	9231	9235	9437	9441	9445	9449	9490	9500	9568
9572	9576	9608	9627	9739	9753	9766	9775	9784	9793	9839	9900	9903
10024	10027	10030	10092	10111	10125	10148	10154	10160	10170	10178	10186	10194
10200	10205	10209	10213	10217	10221	10225	10229	10235	10245	10249	10253	10265
10270	10274	10278	10282	10286	10292	10360	10372	10379	10386	10393	10401	10409
10480	10486	10492	10498	10505	10512	10560	10587	10601	10607	10615	10621	10644
10664	10670	10680	10684	10688	10705	10710	10717	10734	10740	10747	10753	10770
10814	10819	10876	10881	10913	10918	10940	10944	10964	10969	10974	10981	11003
11009	11014	11021	11049	11054	11059	11065	11070	11077	11104	11109	11115	11120
11127	11155	11159	11164	11170	11175	11182	11209	11215	11221	11226	11233	11261
11267	11272	11277	11282	11289	11318	11324	11329	11336	11364	11369	11376	11404
11409	11414	11419	11426	11451	11457	11463	11468	11486	11495	11504	11515	11524
11533	11542	11593	11599	11606	11613	11620	11627	11636	11687	11693	11700	11707
11714	11721	11730	11780	11785	11790	11798	11826	11835	11844	11853	11906	11915
11924	11933	11983	11988	11992	12017	12021	12028	12056	12060	12064	12071	12102
12106	12112	12139	12143	12149	12173	12180	12205	12209	12215	12241	12247	12252
12276	12283	12293	12298	12305	12311	12320	12325	12354	12360	12369	12375	12402
12407	12416	12422	12450	12462	12489	12495	12505	12519	12530	12557	12563	12572
12579	12584	12591	12602	12609	12614	12621	12636	12642	12654	12660	12672	12678
12705	12711	12720	12726	12756	12768	12794	12800	12811	12818	12823	12830	12845
12851	12862	12869	12874	12881	12896	12902	12913	12922	12926	12932	12959	12965
12976	12983	12989	13001	13015	13021	13035	13041	13124	13134	13336	13343	13350
13361	13368	13377	13528	13535	13542	13553	13560	13569	13740	13747	13754	13765
13772	13782	13941	13948	13955	13966	13973	13983	14152	14159	14166	14178	14185
14194	14203	14443	14450	14457	14469	14476	14485	14494	14582	14589	14593	14691
14698	14705	14714	14721	14731	14754	14758	14785	14790	14813	14817	14841	14845
14849	14855	14879	14883	14887	14893	14919	14923	14927	14933	14957	14961	14965
14971	14997	15001	15005	15011	15035	15039	15043	15049	15075	15079	15083	15089
15116	15121	15129	15301	15308	15315	15429	15436	15443	15696	15703	15710	15722

```
                    15729  15739  15921  15928  15935  15944  15951  15961  16098  16103  16110  16160  16166
                    16172  16213  16306  16310  16314  16318  16324  16328  16331  16333  16344  16346  16348
                    16396  16401  16468  16590  16596  16600  16606  16609  16678  16688  16694  16778  16819
                    16824  16829  16834  16913  16923  16927  16931  16939  16943  16948  16955  17007  17014
                    17076  17087  17093  17096  17154  17162  17167  17171  17208  17211  17223  17332  17349
                    17360  17375  17385  17475  17487  17529  17533  17540  17544  17549  17553  17633  17636
                    17658  17661  17770  17774  17826  17957  18200  18358  18370  18380  18432  18437  18446
                    18454  18460  18469  18536  18541  18544  18567  18575  18578  18619  18625  18673  18690
                    18694  18757  18769  18775  18778  18781  18785  18799  18840  18845  18853  18911  18916
                    18934  18938  18947  19005  19008  19088  19092  19096  19120  19123  19126  19174  19205
                    19209  19246  19268  19274  19278  19307  19351  19355  19359  19362  19376  19401  19475
                    19489  19556  19571  19577  19592  19642  19646  19659  19902  19912  19958  20002  20011
                    20059  20062  20077  20080  20091  20106  20113  20116  20144  20147  20161  20175  20178
                    20183  20244  20272  20321  20363  20422  20518  20544  21424  21427  21430  21555  21612
                    21635
ERROUT  101652      16154  16176#
ERRVEC- 000004        513#   1460   1461*  1472*  1479*  1480*  18668  18669*  18670*  18676*  18849  18850*  18851*
                    18856*  19167  19168*  19169*  19177*  19779  19780*  19781*  19814*  19866  19867*  19868*  19881*
                    21447  21448*  21449*  21493*  21742  21743*  21745*  21748*
ERTYPE  131144      21244#  21820
ET    - ****** U     7381
ETO   - ****** U     7408
EXBAD   110126      18010  18020  18064#
EXBAD2  111234      18252  18260  18265  18308#
EXITST  110136      18062  18066#  18187  18193
EXPBDT  105400      17339  17403#
EXPDAT  110150       5123*  5133*  5146   5457*  5458   5465   5469   5968*  5969*  5986   6231*  6250   6254
                    18071#
EXPIR1  034136       7759   7770#
EXPTBL  102564      16449  16486#
EXPWDT  105370      17323  17398#
FACU  - ****** U    10444
FINNOP  021042       5144   5178#
FIN1    052146      10554  10563#
FIN10   053466      10871  10897#
FIN11   053562      10922  10928#
FIN116  022532       5539   5591#
FIN117  022646       5621   5626#
FIN120  022770       5641   5657   5660#
FIN121  023366       5750#
FIN122  023636       5668   5777   5826#
FIN125  025000       5991   6029   6033   6125#
FIN126  026040       6265   6388#
FIN127  026500       6445   6449   6552#
FIN13   053746      10979  10985#
FIN130  027370       6631   6635   6791#
FIN14   054076      11019  11025#
FIN15   054264      11075  11081#
FIN16   054430      11125  11131#
FIN17   054612      11180  11186#
FIN2    052242      10578  10590#
FIN20   054764      11231  11237#
FIN21   055154      11287  11293#
FIN22   055320      11334  11340#
FIN23   055440      11374  11379#
FIN24   055614      11424  11430#
```

```
FIN26   056274          11537  11571#
FIN27   056602          11631  11665#
FIN30   057112          11725  11759#
FIN31   057232          11796  11802#
FIN32   057504          11848  11882#
FIN33   057760          11928  11962#
FIN4    052562          10660  10691#
FIN5    052652          10715  10720#
FIN6    053026          10759  10773#
FIN7    053246          10809  10835#
FLAG    002776           12334#  7781#  7789   7794#  7987#  8005#  8258#  8331#  8517#  8548#  8569#  8671#  8798#
                         8910#  9022#  9048#  9060#  9080#  9084   9108   9127#  9138#  9142   9168#  9208#  9215
                         9220   9237#  9378#  9386#  9397#  9408#  9421#  9435   9460#  9485#  9488   9494#  9498
                         9502#  9509#  9520#  9527#  9534#  9541#  9548#  9555#  9566   9588#  9824#  10039#  10136#
                        10239#  10243  13058#  13067#  13076#  13126  13153#  13163#  13172#  13180#  13189#  13197#  13205#
                        13213#  13221#  13229#  13237#  13245#  13253#  13262#  13270#  13279#  13288#  13297#  13334   13341
                        13370   13394#  13406#  13418#  13429#  13440#  13451#  13462#  13473#  13485#  13526   13533   13562
                        13586#  13594#  13602#  13610#  13618#  13626#  13634#  13642#  13650#  13658#  13666#  13674#  13683#
                        13692#  13701#  13738   13745   13774   13798#  13809#  13820#  13831#  13842#  13853#  13864#  13875#
                        13887#  13898#  13939   13946   13975   13999#  14008#  14017#  14026#  14035#  14045#  14054#  14063#
                        14072#  14081#  14090#  14099#  14108#  14150   14157   14196   14223#  14236#  14249#  14262#  14275#
                        14288#  14301#  14314#  14327#  14340#  14354#  14367#  14381#  14395#  14441   14448   14487   14611#
                        14619#  14627#  14635#  14643#  14651#  14689   14696   14723   15462#  15472#  15482#  15492#  15502#
                        15512#  15523#  15534#  15545#  15556#  15567#  15577#  15587#  15598#  15609#  15619#  15630#  15640#
                        15651#  15694   15701   15731   15757#  15765#  15773#  15781#  15789#  15797#  15805#  15813#  15821#
                        15829#  15837#  15845#  15853#  15861#  15869#  15877#  15919   15926   15953   15979#  15987#  15995#
                        16003#  21610   21629#
FLO     003022           12409#  8035#  8036   8068   8106   8117   8128   8139   8168
FLOAT   003012           12399#  8022   8027   8069   8080   8093   8167   8183   8208   8221   8234
FMPARR  102230          16291  16355#
FPPTST= ****** U         1280   10299  21632
FPTS1 = ****** U        10541
FPTS10= ****** U        10837
FPTS11= ****** U        10899
FPTS12= ****** U        10930
FPTS13= ****** U        10948
FPTS14= ****** U        10987
FPTS15= ****** U        11027
FPTS16= ****** U        11083
FPTS17= ****** U        11133
FPTS2 = ****** U        10565
FPTS20= ****** U        11188
FPTS21= ****** U        11239
FPTS22= ****** U        11295
FPTS23= ****** U        11342
FPTS24= ****** U        11381
FPTS25= ****** U        11432
FPTS26= ****** U        11472
FPTS27= ****** U        11573
FPTS3 = ****** U        10592
FPTS30= ****** U        11667
FPTS31= ****** U        11761
FPTS32= ****** U        11804
FPTS33= ****** U        11884
FPTS4 = ****** U        10625
FPTS5 = ****** U        10693
```

```
FPTS6 = ****** U        10722
FPTS7 = ****** U        10775
FPVEC = 000244           12740  12568* 12596* 12628* 12808* 12835* 12856* 12886* 12907* 13321* 13513* 13725* 13926*
                        14135* 14426* 14566* 14676* 14836* 14874* 14914* 14952* 14992* 15030* 15070* 15108* 15285*
                        15413* 15678* 15902* 16082*
FRSTST  004536           15200
FSRCMO= ****** U        11964
FSTADD  110166          17981* 17984* 17986  18041  18078# 18231* 18234* 18237  18283
FWDSEQ  110154          17969* 17979  18043  18055  18057* 18059* 18073# 18224* 18229  18285  18299  18301* 18303*
GNS   = ****** U          532   1507  22392  22393  22394  22395  22396  22398  22400  22401  22402  22403
GOODAD  002770           12300  17752* 17753* 17768  17772  17779*
GPR0TS  004700           16020
GPR1TS  004756           16260
GPR2TS  005034           16500
GPR3TS  005112           16740
GPR4TS  005170           16980
GPR5TS  005246           17220
GPR6TS  005324           17460
GTSWR = 104406            1502  22398#
HITMIS= 177752           11940  16168  16171  16299  16337  16393  16398  16450  16463  16587  16593  16603  16675
                        16685  16691  16755  16769  16816  16821  16826  16831  16920  16936  17004  17526  17537
                        17546  17630  17655  17765  18249  18255  18262  19798  20056  20074  20088  20110  20141
                        20180
HMPARR  102366          16387  16405#
HOP10   065454          13307  13381#
HOP11   066416          13498  13573#
HOP12   067424          13711  13785#
HOP13   070422          13911  13986#
HOP14   071462          14118  14209#
HOP15   073122          14410  14498#
HOP16   073422          14552  14597#
HOP17   074054          14660  14735#
HOP18   076054          15268  15318#
HOP19   076456          15397  15447#
HOP20   100046          15663  15743#
HOP21   101102          15886  15965#
HOP22   101530          16068  16115#
HOP44   064322          13092  13137#
HT    = 000011            4230  21980  22038
IALL  = ****** U         7557
IL    = ****** U         7503
ILAOA   030552           7024   7031#
ILBOB   030560           7026   7034#
ILL     052646          10697  10717#
ILLA  = ****** U         7018
ILLB  = ****** U         7042
ILLBOA  030646           7047   7054#
ILLBOB  030654           7049   7057#
ILLO  = ****** U         6905
ILLOP1  052502          10630  10662#
ILLOP2  052556          10634  10688#
ILO   = ****** U         7524
INITMM  131362          16666  16743  17611  17727  17927  18171  19782  20123  21322#
INQ22   120530          20254#
INTERR  104476          17138  17144  17159#
INTRPC  104454          17151# 17165
```

```
IOT    = ******  U          7330
IOTIME= ******  U          7613
IOTO   = ******  U          6928
IOTOA    030160            6934    6940#
IOTOB    030166            6936    6943#
IOTVEC= 000020              518#    1444*   1445*
IOXXX    033312            7613#
ITO    = ******  U          7350
JMP    = ******  U          3473
JP     = ******  U          3666
JSI    = ******  U          7580
JSRA   = ******  U          3715    3862
KDPAR0= 172360              406#   17972   17974*  18214   18216*
KDPAR1= 172362              407#
KDPAR2= 172364              408#
KDPAR3= 172366              409#
KDPAR4= 172370              410#
KDPAR5= 172372              411#
KDPAR6= 172374              412#
KDPAR7= 172376              413#
KDPDR0= 172320              384#
KDPDR1= 172322              385#
KDPDR2= 172324              386#
KDPDR3= 172326              387#
KDPDR4= 172330              388#
KDPDR5= 172332              389#
KDPDR6= 172334              390#
KDPDR7= 172336              391#
KIPAR0= 172340              395#   9811*
KIPAR1= 172342              396#
KIPAR2= 172344              397#
KIPAR3= 172346              398#
KIPAR4= 172350              399#
KIPAR5= 172352              400#   18218*  18247*  18253*  18308   18310*
KIPAR6= 172354              401#    9729*   9745*   9761*   9770*   9779*   9788*   9813*  10012*  16744*  17464*  17612*  17732*
                          17736   17745*  17746   17748*  17756*  17758*  17781*  17928*  18186*  19783*  19792*  19806   20124*
                          21214   21230
KIPAR7= 172356              402#
KIPDR0= 172300              373#    9812*  16669*  16673*  16679*  16682*  17971*  17975*  18213*  18217*
KIPDR1= 172302              374#
KIPDR2= 172304              375#
KIPDR3= 172306              376#
KIPDR4= 172310              377#
KIPDR5= 172312              378#   18212*
KIPDR6= 172314              379#    9816*  17738*  17740*  20125*
KIPDR7= 172316              380#
KMCR   = 177734            1195#
LASTCH   112060           18553#
LCD    = ******  U         15320
LCDSUB   076344           15332   15339   15346   15353   15360   15367   15374   15381   15388   15412#
LCF    = ******  U         15135
LCFSUB   075742           15147   15153   15159   15165   15171   15177   15183   15189   15195   15201   15207   15213   15219
                          15225   15231   15237   15243   15249   15255   15261   15284#
LDC    = ******  U         12937
LDC2   = ******  U         15094
LDDM2  = ******  U         11997
```

```
LDDM3 = ****** U        12033
LDDM4 = ****** U        12077
LDDM5 = ****** U        12118
LDDM6 = ****** U        12153
LDDM7 = ****** U        12184
LDM27 = ****** U        12219
LDPARS  131524          21323   21325   21327   21329   21331   21333   21367#
LDPDRS  131554          21335   21337   21339   21341   21343   21345   21396#
LEDS    116226          19686   19710#
LF    = 000012            424#  22032   22038
LKS   = 177546           1196#  18671   18685   18688   18754*  18755   18766*  18767   18770   18772*  18773   18776   18792*
                        18852   18945   18948*  18994*  19003   19006   19069*  19084*
LKSFL   002702           1178#  18680*  18692   18751*  18759   18782   18789*  18793   18796   18925*  18928   18935   18995*
                        18998   19068*  19073   21222   21433*
LKSINT  131650          18681   18752   18920   18992   19058   21433#
LOOP    004534           1487    15110#  21716
LOOPIN  003114           1253#  16446*  16451   16458   16470*  17730*  17788*
LOST    052132          10545   10547   10549   10551   10556#
LOWADD  002766           1229#  17726*  17734   17752   17787*
LSTADD  110170          17982*  17985*  18021   18079#  18232*  18235*  18266
LS1   = ****** U        14822
LS2   = ****** U        14860
LS3   = ****** U        14898
LS4   = ****** U        14938
LS5   = ****** U        14976
LS6   = ****** U        15016
LS7   = ****** U        15054
LXP   = ****** U        15449
LXPSUB  077626          15463   15473   15483   15493   15503   15513   15524   15535   15546   15557   15568   15578   15588
                        15599   15610   15620   15631   15641   15652   15677#
MACCC   016006           4289#
MACE    051114          10334   10337   10340   10343   10352#
MACO    051004          10330   10332#
MACOA   051010          10333#  10347   10351
MAC1    051020          10335#
MAC2    051034          10338#
MAC3    051050          10341#
MAC4    051064          10344#
MAC5    051100          10348#
MADCC   015732           4260#
MAIREG= 177750           1197#  19118   19121   19124   21441
MALCC   016560           4519#
MARCC   016640           4551#
MARKIT= ****** U         4754
MASK    003120           1257#
MA11    010604           2908#
MA55    011510           3169#
MB811   010702           2944#
MB822   011164           3053#
MBCCC   015454           4143#
MBC00   010406           2831#
MBC11   010762           2974#
MBC22   011252           3082#
MBI00   010326           2801#
MBPT0   030406           6995#
MBSCC   015526           4170#
```

```
MBT      032516            7449#
MBTA     032540            7456#   7462
MBTB     032542            7454    7458#
MBTCC    015404            4117#
MBTD     032552            7459    7462#
MBTE     051450           10425   10441#
MBTF     032562            7463    7466#
MBTO     032574            7471#
MBTOA    032624            7480#
MBTOB    032626            7476    7482#
MBTOC    032640            7483    7486#
MBTOD    032664            7491#
MBTOE    032666            7487    7493#
MBTOF    032700            7494    7497#
MBT1     050732           10305#
MBT2     050734           10315#
MBT2A    050746           10318#  10426
MBT8     051156           10359   10363#
MBT8A    051206           10366   10368   10371   10374#
MBT8B    051230           10375   10378   10381#
MBT8C    051252           10382   10385   10388#
MBT8D    051274           10389   10392   10395#
MBT8E    051320           10396   10400   10403#
MBT8F    051344           10404   10408   10411#
MBT8FG   051360           10414   10416#
MBT8I    051404           10420   10423#
MB66     011564            3200#
MCB44    011430            3141#
MCLRD    074110           14771#
MCLRI    074166           14803#
MCMPD    064022           13054#
MCTSCC   015656            4227#
MDAO     010202            2737#
MDCCC    015574            4195#
MDDSUB   072656           14224   14237   14250   14263   14276   14289   14302   14315   14328   14341   14355   14368   14382
                           14396   14425#
MDFSUB   071206           14000   14009   14018   14027   14036   14046   14055   14064   14073   14082   14091   14100   14109
                           14134#
MDIVD    065454           13390#
MDIVF    064322           13149#
MDMO     010156            2721#
MDM27    010266            2779#
MDSO     010232            2757#
MD1    = ******  U         1520
MD2    = ******  U         1542
MD3    = ******  U         1557
MD4    = ******  U         1570
MD5    = ******  U         1583
MD6    = ******  U         1592
MEMK     116630           19822   19832#
MEMQ     116660           19828   19836#
MEMTO    030212            6951#
MET      032234            7381#
META     032272            7390#   7399
METB     032302            7386    7395#
METD     032312            7396    7399#
```

J3


```
METF    032322            7400    7403#
METO    032342            7408#
METOA   032406            7419#
METOB   032416            7413    7424#
METOC   032430            7425    7428#
METOD   032454            7433#
METOE   032464            7429    7438#
METOF   032476            7439    7442#
ME100 - ****** U          4809
ME101 - ****** U          4820
ME102 - ****** U          4830
ME103 - ****** U          4840
ME104 - ****** U          4850
ME105 - ****** U          4860
ME106 - ****** U          4870
ME107 - ****** U          4880
ME110 - ****** U          4890
ME111 - ****** U          4900
ME112 - ****** U          4910
ME113 - ****** U          4975
ME113A- ****** U          5039
ME114 - ****** U          5113
ME115 - ****** U          5416
ME116 - ****** U          5495
ME117 - ****** U          5593
ME120 - ****** U          5628
ME121 - ****** U          5664
ME122 - ****** U          5752
ME123 - ****** U          5828
ME124 - ****** U          5893
ME125 - ****** U          5963
ME126 - ****** U          6127
ME127 - ****** U          6390
ME130 - ****** U          6554
MFA     051452          104554#
MFACU   051450          104444#
MFSRCH  057762          119724#
MIALL   033112            7557#
MIALLA  033136            7565#   7571
MIALLB  033140            7562    7567#
MIALLD  033150            7568    7571#
MIALLF  033160            7572    7575#
MIL     032712            7503#
MILA    032736            7511#   7517
MILAO   030504            7018#
MILB    032740            7508    7513#
MILD    032750            7514    7517#
MILF    032760            7518    7521#
MILLBO  030604            7042#
MILLO   030016            6905#
MILLOA  030062            6911    6916#
MILLOB  030070            6913    6919#
MILO    032772            7524#
MILOA   033024            7534#
MILOB   033026            7529    7536#
MILOC   033040            7537    7540#
```

MILOD	033064	7545#		
MILOE	033066	7541	7547#	
MILOF	033100	7548	7551#	
MIOT	032040	7330#		
MIOTA	032062	7337#	7343	
MIOTB	032064	7335	7339#	
MIOTD	032074	7340	7343#	
MIOTF	032104	7344	7347#	
MIOTO	030114	6928#		
MITO	032116	7350#		
MITOA	032146	7359#		
MITOB	032150	7355	7361#	
MITOC	032162	7362	7365#	
MITOD	032206	7370#		
MITOE	032210	7366	7372#	
MITOF	032222	7373	7376#	
MJ	012514	3473#		
MJP	013346	3666#		
MJP17	013356	3672#		
MJP27	013400	3682#		
MJP27A	013412	3683	3686#	
MJP37	013464	3687	3704#	
MJP37A	013476	3705	3708#	
MJP67	013422	3688#	3709	
MJP67A	013434	3689	3692#	
MJP67B	013450	3694	3698#	
MJP77	013452	3698	3700#	
MJP77E	013506	3701	3712#	
MJRA	014264	3862#		
MJR27	014320	3872#	3878	
MJR27A	014354	3879	3882#	
MJR27B	014374	3886#	3911	
MJR37	014456	3885	3905#	
MJR37A	014512	3912	3915#	
MJR6A	014452	3902#	3926	
MJR6B	014454	3901	3903#	
MJR67	014376	3888#	3918	
MJR67A	014432	3895	3898#	
MJR77	014532	3894	3903	3920#
MJR77A	014566	3927	3930#	
MJSI	033172	7580#		
MJSIA	033224	7590#		
MJSIB	033226	7585	7592#	
MJSIC	033240	7593	7596#	
MJSID	033264	7601#		
MJSIE	033266	7597	7603#	
MJSIF	033300	7604	7607#	
MJSR	013506	3715#		
MJSRA	013624	3741	3745#	3795
MJSRB	013770	3781	3784#	3835
MJSRC	014136	3822	3824#	
MJSR1	013626	3722	3747#	
MJSR1A	013640	3748	3751#	
MJSR1B	013664	3756	3759#	
MJSR2	013544	3727#	3755	3762
MJSR2A	013556	3728	3731#	

L3


```
MJSR2B  013602          3736    3739#
MJSR3   013772          3745    3787#
MJSR3A  014004          3788    3791#
MJSR3B  014030          3796    3799#
MJSR4   013706          3735    3766#   3802
MJSR4A  013720          3767    3770#
MJSR4B  013744          3775    3778#
MJSR5   014140          3784    3827#   3854
MJSR5A  014152          3828    3831#
MJSR5B  014176          3836    3839#
MJSR6   014054          3774    3807#   3842
MJSR6A  014066          3808    3811#
MJSR6B  014112          3816    3819#
MJSR7   014222          3815    3824    3846#
MJSR7A  014234          3847    3850#
MJSR7E  014260          3855    3858#
MJU1    012566          3477    3491#
MJU1A   012600          3492    3495#
MJU2    012530          3479#   3483    3496
MJU2A   012542          3480    3483#
MJU2B   012552          3484    3487#
MJU3    012612          3490    3498#
MJU3A   012624          3499    3502#
MJU3B   012634          3503    3506#
MJU4    012706          3507    3522#   3523
MJU4A   012720          3524    3527#
MJU4B   012732          3528    3531#
MJU5    012646          3509#   3535
MJU5A   012660          3511    3514#
MJU5B   012672          3515    3518#
MJU6    012746          3519    3537#
MJU6A   012760          3538    3541#
MJU7    013000          3544    3546#
MJU7E   013012          3547    3550#
MJ2     012564          3488    3490#   3502
MJ5     012744          3510    3532    3535#
MJ7     012774          3542    3544#
MLCD    076054        15328#
MLCF    075236        15143#
MLDC    063454        12945#
MLDC2   075136        15102#
MLDDM2  060032        12005#
MLDDM3  060120        12042#
MLDDM4  060234        12085#
MLDDM5  060340        12126#
MLDDM6  060430        12161#
MLDDM7  060514        12192#
MLDM27  060602        12227#
MLDSUB  070210        13799   13810   13821   13832   13843   13854   13865   13876   13888   13899   13925#
MLFSUB  067212        13587   13595   13603   13611   13619   13627   13635   13643   13651   13659   13667   13675   13684
                      13693   13702   13724#
MLS1    074222        14830#
MLS2    074314        14868#
MLS3    074406        14906#
MLS4    074516        14946#
MLS5    074612        14984#
```

```
MLS6     074722           15024#
MLS7     075022           15062#
MLXP     076456           15458#
MMARK    017344            4754#
MMODD    071462           14219#
MMODF    070422           13995#
MMRL5    012246            3370#
MMR0   = 177572            1208#  19786#  19815#  20126#  20153#
MMR1   = 177574            1209#
MMR2   = 177576            1210#
MMR3   = 172516            1211#  17729#  17794#  18173#  18207#  19787#  19816#
MMTS10= ****** U           9375
MMTS11= ****** U           9457
MMTS12= ****** U           9506
MMTS13= ****** U           9583
MMTS14= ****** U           9801
MMTS15= ****** U          10036
MMTS16= ****** U          10133
MMTS6A= ****** U           8566
MMTS6B= ****** U           8668
MMTS6C= ****** U           8795
MMTS6D= ****** U           8907
MMU      132344            8577   8677   8804   8916   9028   9046   9133   9173   9382   9394   9405   9419   9481
                           9511   9590   9825  10011  10040  10138  21564#
MMULD    067424           13794#
MMULF    066416           13582#
MMUTRP   132516            1483  21610#
MMUTST= ****** U           7977  21558
MMUTS0= ****** U           3963
MMUTS1= ****** U           7983
MMUTS2= ****** U           8001
MMUTS3= ****** U           8255
MMUTS4= ****** U           8328
MMUTS5= ****** U           8514
MMUTS6= ****** U           8545
MMUTS7= ****** U           9019
MMUTS8= ****** U           9124
MMUTS9= ****** U           9165
MMVCC    015340           40930#
MMVEC  = 000250             274#
MM11     010474            2865#
MM22     011042            3003#
MNCCCC   016070            4321#
MNGOP    061720           12544#
MNNRM1   060674           12263#
MNNRM2   061114           12338#
MNNRM3   061246           12388#
MNNRM4   061370           12435#
MNRM     062552           12739#
MODD   = ****** U          14211
MODE1    045626            9599   9686#
MODE2    047206            9849   9961#
MODF   = ****** U          13987
MODGAR   071452           14139  14207#  14430
MRLB1    011774            3269#
MRLCC    016342            4426#
```

```
MRL0     011722            3244#
MRL2     012054            3297#
MRL3     012132            3325#
MRL4     012206            3350#
MRL5   - ****** U          3370
MRL6     012310            3390#
MRL7     012346            3408#
MRRB1    012466            3456#
MRRCC    016422            4459#
MRRO     012444            3442#
MRT      031316            7181#
MRTA     031340            7188#   7194
MRTB     031342            7186    7190#
MRTE     031352            7191    7194#
MRTF     031362            7195    7198#
MRTO     031374            7204#
MRTOA    031424            7213#
MRTOB    031426            7209    7215#
MRTOC    031440            7216    7219#
MRTOD    031464            7225#
MRTOE    031466            7220    7227#
MRTOF    031500            7228    7231#
MRTS     014572            3935#
MSB      062430           12692#
MSBCC    016206            4368#
MSBCCC   016256            4393#
MSCD     100046           15753#
MSCF     101102           15975#
MSDF     073422           14607#
MSER   - 177744            1198#  16162  16165  16356  16405  16616  16910*  16911  16925  16928*  16929  16941  16999*
                          17009*  17079*  17083  17090  17168  17209  17215*  17324*  17329  17340*  17346  17351*  17357
                          17366*  17372  17376*  17382  17387*  19909  19914*  19949  19959*  20004  20005*  20012*  21198
                          21206   21208  21422  21425  21428  21431*
MSFD     073122           14507#
MSFDI    074054           14744#
MSOB     017270            4727#
MSPAA    007434            2479#
MSPAU    027370            6792#
MSPB     005526            1813#
MSPBB    007520            2513#
MSPC     005552            1831#
MSPD     005602            1850#
MSPEO    005640            1873#
MSPF     005702            1898#
MSPG     005744            1922#
MSPH     006002            1945#
MSPI     006046            1971#
MSPJ     006106            1995#
MSPL     006226            2045#
MSPM     006274            2072#
MSPN     006352            2104#
MSPO     006420            2131#
MSPP     006524            2176#
MSPQ     006622            2211#
MSPR     006730            2252#
MSPS     007016            2283#
```

MSPT	007064	2308#			
MSPU	007102	2322#			
MSPV	007202	2370#			
MSPVO	007146	2348#			
MSPX	007246	2396#			
MSPY	007316	2424#			
MSPZ	007364	2451#			
MSPO	005504	1795#			
MSTB3	007570	2541#			
MSTO	030674	7064#			
MSTOE	030750	7068	7080#		
MSTOEE	030756	7079	7083#		
MST4	007642	2570#			
MST4B	007700	2593#			
MST5	007740	2616#			
MST5B	007772	2637#			
MST6	010034	2662#			
MST7	010102	2687#			
MSW37	012410	3426#			
MSXP	101244	16021#			
MSXT	017072	4648#			
MSXTCC	016730	4586#			
MS11	010642	2926#			
MS22	011114	3030#			
MS33	011344	3116#			
MS77	011634	3220#			
MTP	031512	7235#			
MTPA	031714	7292#			
MTPAA	031770	7300	7301	7307	7312#
MTPAE	032020	7317	7321#		
MTPAH	031766	7299	7309#		
MTPAL	031756	7303#	7318#	7319#	7320
MTPB	031544	7240	7246#		
MTPF	031564	7251	7254#		
MTPO	031576	7260#			
MTPOA	031626	7269#			
MTPOB	031630	7265	7271#		
MTPOC	031642	7272	7275#		
MTPOD	031666	7280#			
MTPOE	031670	7276	7282#		
MTPOF	031702	7283	7286#		
MTPQ	031554	7247	7250#		
MTPR	031542	7244#	7250		
MTRPO	030310	6972#			
MTRY	027532	6836#			
MTRYA	027606	6843	6852#		
MTRYB	027626	6856	6860#		
MTRYM	027660	6870#			
MTSO	015340	4031	4092#		
MTT	030770	7088#			
MTTA	031024	7094	7099#	7105	
MTTB	031026	7095	7101#		
MTTD	031036	7102	7105#		
MTTE	031046	7106	7109#		
MTTR	031152	7142#			
MTTRA	031216	7148	7155#		

```
MTTRB   031220            7149    7157#
MTTRC   031232            7158    7161#
MTTRD   031270            7163    7170#
MTTRE   031272            7164    7172#
MTTRF   031304            7173    7176#
MTTS    031060            7115#
MTTSA   031114            7121    7126#
MTTSB   031120            7122    7130#
MTTSD   031130            7131    7134#
MTTSE   031140            7135    7138#
MTTSQ   031116            7128#   7134
MULD  - ****** U          13786
MULF  - ****** U          13574
MUVAD   062662            12780#
MVCC  - ****** U           4093
MXDF1   061500            12475#
MXOR    017210             4699#
MXRCC   017000             4613#
M11   - ****** U           2865
M2      004562             1542#
M22   - ****** U           3003
M3      004576             1557#
M4      004616             1570#
M5      004636             1583#
M6      004656             1592#
NCCCC - ****** U           4321
NEWADD  002774             1232#
NEWDAT  110162            17978#  18011   18018   18029#  18031   18037#  18039#  18076#  18227#  18253   18268   18272#  18277#
                          18280#
NGOP  - ****** U          12536
NNRM1 - ****** U          12256
NNRM2 - ****** U          12330
NNRM3 - ****** U          12380
NNRM4 - ****** U          12427
NOABRT  133714            21853   21859   21882#
NOQ22   132102            21492   21496#
NOTOK   132574            21618   21624#
NRM   - ****** U          12731
NULL  - 000000             1213#  19206   19390   19483   19653
NXMFIN  047430            10019   10029   10031#
NXMPAR  105570            17473   17486#
NXMTRP  047036             9809    9912#
NXMTST- ****** U          10004
ODDADR- ****** U           7636
ODDXX   033412             7636#
OK      132552            21617   21619#
OKAY7   042650             9094    9097#
OKAY7A  042660             9098    9101#
OKA7    042636             9090    9093#
OK1     132600            21623   21625#
OK7     042620             9085    9088#
PAR     132466            21569   21570   21573   21574   21576   21577   21595#
PARAD1  045456             9592    9634#
PARAD2  047070             9845    9922#
PARVA1  045510             9593    9647#
PARVA2  047122             9844    9935#
```

```
PARVA3  047240            9829    9851    9974#
PAR1    132470          215964  21599
PCR   - 177522            1199#  18344#  18351#  18352   18354   18363#  18364   18366   18376#  18377   18523#  18534   18546#
                         18547#  18565   18569#  18570#  18579#  18609#  18626#  21218
PDR     132444          21568   21572   21575   21586#
PDR1    132446          21587#  21590
PHY1    045574            9597    9673#
PIR   - 177772            1200#
PIRQ  - 177772             430#   7726    7745#   7746    7750#   7751    7780#   7784#   7786    7818#   7829    7830#   7845#
                          7851#   7855#   7858#   7874#   7883#   7886    7903    7904#   7946#   7951#   7961    7970#  20314#
                         20329#
PIRQNX  034014             7724    7730#
PIRQT   121020           20308   20331#
PIRQVE- 000240             524#   7735    7736    7737#   7738#   7814#   7815#   7849#   7850#   7875#   7876#   7948#   7949#
                          7973#   7974#  20305#  20306#
PIRRTN  034314             7814    7829#
PIRTBL  034116             7742    7761#
PIRTEX  034016             7728    7731#
PIRTST- ****** U           7720
PIRXXX  033760             7721#
PIR1    034016             7733#
PIR2    034136             7773#
PIR2EX  034244             7790    7807#
PIR3    034244             7809#
PIR3EX  034340             7827    7837#
PIR4    034340             7840#
PIR5    034416             7864#
PIR5EX  034626             7898    7932#
PIR6    034626             7935#
PIR6EX  034730             7958    7973#
PITBL1  034222             7783    7800#
PITBL2  034612             7877    7925#
PI1     034266             7817#   7826
PI2     034300             7823#   7833
PI3     034302             7824#   7834
PLF0    043466             9174    9251#
PLF1    043660             9185    9196    9315#
POLY  - 120001            1212#
PROCNT  022716           5639#   5652
PR0   - 000000             447#
PR1   - 000040             448#
PR2   - 000100             449#
PR3   - 000140             450#
PR4   - 000200             451#
PR5   - 000240             452#
PR6   - 000300             453#
PR7   - 000340             454#   7738    7876    7949
PS    - 177776             427#    428    5190#   5191    5202#   5203    5231#   5238#   5245#   5413#   5636#   7150#   7165#
                          7210#   7221#   7241#   7266#   7277#   7356#   7367#   7416#   7430#   7477#   7488#   7530#   7542#
                          7586#   7598#   7618#   7669#   7670    7674    7678    7881#   7885   10017#
PSW   - 177776             428#   7698#   7846#
PSWBTS  005406            17710
PWRVEC- 000024             519#   1450#   1451#  20357   20358#  20359#  20368#  22408#  22409#  22418#  22424#  22436#  22437#
Q22EN   002752            1186#  20227   20256   20307   20403
Q22INT  132142          20238   20267   20315   20412   20416   21508#
Q22SIZ  131656             1509   21441#
```

```
RAMPAR  131600           1481   21421#
RBUF  = 177562           1201#  19256   19269   19415   19463   19497   19561   19567   19588   19637   19656   19705   21228
RCSR  = 177560           1202#  19170   19252   19262   19265   19275   19352*  19353   19360   19391*  19410*  19412   19460
                        19484*  19501*  19558   19565   19586   19630   19654   19689*
RDCHR = 104410          22297   22401#
RDLIN = 104411          22338   22402#
RDOCT = 104412          22403#
RECDAT  110152          16393*  16394   16398*  16399   16465   16483*  16587*  16588   16593*  16594   16603*  16604   16675*
                        16676   16685*  16686   16691*  16692   16755*  16756   16769*  16770   16816*  16817   16821*  16822
                        16826*  16827   16831*  16832   16936*  16937   17004*  17005   17526*  17527   17530   17537*  17538
                        17541   17546*  17547   17550   17629*  17634   17654*  17659   17765*  17766   17954*  18007*  18008
                        18017*  18018   18064   18072#  18249*  18250   18255*  18256   18258   18262*  18263   19798*  19799
                        20056*  20057   20074*  20075   20088*  20089   20110*  20111   20141*  20142   20180*  20181   21202
                        21212
RECDST  003102           1249#  12008   12045   12067   12088   12129   12164   12195   12230   12267*  12268*  12278   12341
                        12391   12442   12481   12553   12575   12605   12632   12650   12668   12698   12750   12763   12790
                        12814   12841   12865   12892   12916   12955   12979   13011   13031   13120   13357   13549   13761
                        13962   14173   14464   14570   14680   14778   15124   15286   15414   15679   15903   16083   16101
RECFEC  003062           1247#  11977   14727   15735   15957
RECST   003072           1248#  14835   14843   14847   14873   14881   14885   14913   14925   14951   14959   14963   14991
                        15003
REDZON= ****** U          7691
RESVEC= 000010            5140#
RET1    015056           4015#   4047    4062    4085
RET2    015144           4037#   4067    4080
RET3    015222           4057#   4075
RITEDA  110160          17977*  18014   18016   18023   18028*  18033*  18034*  18037   18039   18075#  18228*  18271*  18272
                        18276*  18279*  18280
RLB1  = ****** U          3269
RLCC  = ****** U          4426
RLO   = ****** U          3244
RL2   = ****** U          3297
RL3   = ****** U          3325
RL4   = ****** U          3350
RL6   = ****** U          3390
RL7   = ****** U          3408
ROM   = ****** U          3552
RRB1  = ****** U          3456
RRCC  = ****** U          4459
RRO   = ****** U          3442
RT    = ****** U          7181
RTO   = ****** U          7204
RTS   = ****** U          3935
RTSE    014644           3958   3961#
RTS1    014612           3940   3944#
RTS6    014622           3945   3951#
RXXX    033632           7691#
SAVBR   002710           1181#  18420*  18471   18517*  18580*  18601*  18627   18835*  18857   18906*  18949   18990*  19057*
                        19097   19551*  19582   19942*  19961   20498*  20545
SAVMR0  003004           1236#   9025*   9056*   9088*   9089    9101*   9112*   9113    9130*   9146*   9147    9169*   9224*
                         9225   9238*   9379*   9439    9451*  10251   21625*
SAVMR1  003006           1237#   9026*   9057*   9093    9102*   9117    9131*   9151    9170*   9229    9239*   9380*   9443
                         9452*  9510*   9570    9578*  21626*
SAVMR2  003010           1238#   9027*   9058*   9097    9103*   9132*   9155    9171*   9233    9240*   9381*   9447    9453*
                         9574   9579*  10247   21627*
SAVPCR  002706           1180#
```

```
SAVPOS  003116          1256#
SAVSUP  003000          1234#  8575*  8597   8675*  8697   8802*  8825   8914*  8937
SAVUSE  003002          1235#  8572*  8592   8673*  8692   8800*  8820   8912*  8932
SAV30   004052          1414   1424*  1427#  21865  21875
SAV32   004054          1425*  1428#  21866  21876
SB    - ****** U        12684
SBCC  - ****** U         4368
SBCCC - ****** U         4393
SCD   - ****** U        15745
SCDSUB  100666          15758  15766  15774  15782  15790  15798  15806  15814  15822  15830  15838  15846  15854
                        15862  15870  15878  15901# 15980  15988  15996  16004
SCF   - ****** U        15967
SCOPE - 000004           4204#  1513   16151  16203  16289  16385  16445  16519  16665  16742  16811  16907  16996
                        17057  17136  17192  17318  17460  17521  17610  17723  17811  17923  18167  18343  18419
                        18516  18600  18660  18743  18827  18898  18978  19046  19117  19158  19198  19238  19295
                        19339  19450  19539  19625  19676  19768  19862  19933  19978  20038  20216  20298  20352
                        20390  20462  20549
SDF   - ****** U        14599
SDFSUB  073652          14612  14620  14628  14636  14644  14652  14675#
SDPAR0- 172260            362#
SDPAR1- 172262            363#
SDPAR2- 172264            364#
SDPAR3- 172266            365#
SDPAR4- 172270            366#
SDPAR5- 172272            367#
SDPAR6- 172274            368#
SDPAR7- 172276            369#
SDPDR0- 172220            340#
SDPDR1- 172222            341#
SDPDR2- 172224            342#
SDPDR3- 172226            343#
SDPDR4- 172230            344#
SDPDR5- 172232            345#
SDPDR6- 172234            346#
SDPDR7- 172236            347#
SEQ     003032          1241#  3476*  3479   3487*  3491   3495*  3498   3506*  3514   3518*  3527   3531*  3537
                        3541*  3546   3669*  3672   3676*  3682   3686*  3688   3692*  3700   3704   3708*  3721*
                        3727   3739*  3747   3759*  3766   3778*  3787   3799*  3807   3819*  3827   3839*  3846
                        3865*  3872   3882*  3888   3898*  3905   3915*  3920
SFD   - ****** U        14499
SFDI  - ****** U        14736
SFDSUB  073316          14511  14519  14527  14535  14543  14565#
SIMGOA  002662          1167#  20483* 20540* 21466* 21467*
SIPAR0- 172240            351#
SIPAR1- 172242            352#
SIPAR2- 172244            353#
SIPAR3- 172246            354#
SIPAR4- 172250            355#
SIPAR5- 172252            356#
SIPAR6- 172254            357#
SIPAR7- 172256            358#
SIPDR0- 172200            329#  10237*
SIPDR1- 172202            330#
SIPDR2- 172204            331#
SIPDR3- 172206            332#
SIPDR4- 172210            333#
```

```
SIPDR5= 172212            334#
SIPDR6= 172214            335#
SIPDR7= 172216            336#
SIXBIT  111426          18345   18346   18384#
SLEND   116044          19163   19662#
SLOC00  002762           1225#   5122*   5129    5139    6842*   6862    6875*   6900    6910*   6921    6933*   6947    6956*
                         6967    6977*   6988    7000*   7012    7023*   7036    7046*   7059    7067*   7083    7092*   7110
                         7119*   7139    7146*   7177    7185*   7199    7208*   7232    7239*   7255    7264*   7287    7297*
                         7323    7334*   7347    7354*   7377    7385*   7404    7412*   7444    7453*   7466    7475*   7498
                         7507*   7521    7528*   7552    7561*   7575    7584*   7608    7616*   7634    7643*   7688    7694*
                         7715    7723*   7727    7735*   7973    9522*   9529*   9536*   9543*   9550*   9557*   9574    9724*
                         9798    9807*   9821    9914    10013*  10033   16153*  16173   16290*  16349   16386*  16402   16580*
                        16611   16908*  16945   16950   16997*  17008   17058*  17077   17137*  17155   17200*  17226   17319*
                        17388   17461*  17480
SLOC01  002764           1226#   6912*   6920    6935*   6946    6958*   6968    6979*   6989    7002*   7013    7025*   7035
                         7048*   7058    7298*   7324    7387*   7403    7414*   7443    7645*   7689    7736*   7974    9725*
                         9799    9808*   9822    9915    17059*  17078   17462*  17481
S08     = ****** U       4727
SOPAA   = ****** U       2479
SOPBB   = ****** U       2513
SOPD    = ****** U       1850
SOPE    = ****** U       1873
SOPF    = ****** U       1898
SOPG    = ****** U       1922
SOPH    = ****** U       1945
SOPI    = ****** U       1971
SOPJ    = ****** U       1995
SOPL    = ****** U       2045
SOPM    = ****** U       2072
SOPN    = ****** U       2104
SOPO    = ****** U       2131
SOPP    = ****** U       2176
SOPQ    = ****** U       2211
SOPR    = ****** U       2252
SOPS    = ****** U       2283
SOPT    = ****** U       2308
SOPU    = ****** U       2322
SOPV    = ****** U       2370
SOPV0   = ****** U       2348
SOPX    = ****** U       2396
SOPY    = ****** U       2424
SOPZ    = ****** U       2451
SOP1    = ****** U       1813
SOP2    = ****** U       1831
SPAU    = ****** U       6792
SPAU1   027406           6798    6801#
SPAU2   027430           6805    6808#
SPAU3   027450           6811    6814#
SPAU4   027474           6818    6821#
SPAU5   027510           6824    6827#
SPAU6   027526           6830    6833#
SPS     003034           1242#   3718*   3740    3760    3779    3800    3820    3840    3859    3866*   3884    3899    3916
                         3932
SPSJ    003036           1243#   3719*   3720*   3731    3751    3770    3791    3811    3831    3850    3867*   3868*   3874
                         3890    3907    3922
SPT38   = ****** U       2541
```

```
SP0     = ****** U      1794
SR0     = 177572         278#    1208    5694    8518    8525    8532    8539    9730#   9751#   9765#   9774#   9783#   9792#
                        9795#   9805#   9818#   9912#   10015#  10031#  16670#  16672#  16680#  16683#  16689#  16695#  16746#
                       16775#  17465#  17479#  17613#  17664#  17728#  17792#  17929#  17963#  18172#  18206#
SR1     = 177574         279#    1209
SR2     = 177576         280#    1210
SR3     = 172516         281#    1211    9728#   9744#   9758#   9817#
STACK   = 001100         418#    1442
START     003764         537      540    1411#
STBOT   = 001000        1277#    1768    3938    4010    4013#   4026    4757    4758#   4775    4799    6001    6014    6016
                        6137    6142    6149    6154    6808    6814    6827    6833    6849    6866    6879    6888    6896
                        6901    6916    6923    6940    6945    6963    6970    6984    6991    7007    7014    7031    7038
                        7054    7060    7080    7084    7091    7101    7112    7118    7130    7140    7145    7157    7161
                        7172    7178    7184    7190    7200    7207    7215    7219    7227    7233    7238    7246    7256
                        7263    7271    7275    7282    7288    7296    7315    7326    7333    7339    7348    7353    7361
                        7365    7372    7378    7384    7395    7405    7411    7424    7428    7438    7445    7452    7458
                        7467    7474    7482    7486    7493    7499    7506    7513    7522    7527    7536    7540    7547
                        7553    7560    7567    7576    7583    7592    7596    7603    7609    7714    8571    8574    9740
                        9797
STKLMT= 177774           429#
STMOVI  107330          17942   17967#
STMOVT  110416          18192   18211#
ST0     = ****** U      7064
ST4     = ****** U      2570
ST48    = ****** U      2593
ST5     = ****** U      2616
ST58    = ****** U      2637
ST6     = ****** U      2662
ST7     = ****** U      2687
SUBT      051740        10465   10468   10470   10472   10474   10516#
SWR       001140          601#    1440    1462#   1464    1470#   1477#   1500    9707   17924   18168   21737   21751   21753
                        21761   21768   21811   21818   21830   21834   22192   22229#  22416   22429#
SWREG     000176         534#    1470    1500   22192   22205
SW0     = 000001         482#    3426
SW00    = 000001         472#     482
SW01    = 000002         471#     481
SW02    = 000004         470#     480
SW03    = 000010         469#     479
SW04    = 000020         468#     478
SW05    = 000040         467#     477
SW06    = 000100         466#     476
SW07    = 000200         465#     475
SW08    = 000400         464#     474
SW09    = 001000         463#     473
SW1     = 000002         481#    3426
SW10    = 002000         462#
SW11    = 004000         461#
SW12    = 010000         460#
SW13    = 020000         459#
SW14    = 040000         458#
SW15    = 100000         457#
SW2     = 000004         480#
SW3     = 000010         479#
SW37    = ****** U      3426
SW4     = 000020         478#
SW5     = 000040         477#
```

```
SW6    = 000100            476#
SW7    = 000200            475#
SW8    = 000400            474#
SW9    = 001000            473#
SXP    = ****** U        16013
SXPSUB  101424           16025   16032   16039   16046   16053   16060   16081#
SXT    = ****** U         4648
SXTCC  = ****** U         4586
S11    = ****** U         2926
S22    = ****** U         3030
S33    = ****** U         3116
S77    = ****** U         3220
TAB1     003174           1280#  12009   12019   12023   12131   12145   12197   12211   12393
TAB10    003324           1322#  12482   12515
TAB11    003334           1326#  12480
TAB11A   003344           1330#  12491
TAB12    003354           1332#  12497
TAB13    003364           1334#
TAB13B   003374           1338#  12500
TAB14    003404           1340#  12508   12713
TAB15    003414           1344#  12509
TAB16    003424           1348#  12521   12971
TAB17    003434           1352#  12526
TAB18    003444           1355#  12418
TAB2     003204           1284#  12047   12066   12165   12343
TAB21    003454           1359#  12549   12566   12595   12697
TAB22    003464           1361#  12559
TAB23    003474           1363#  12567   12594
TAB24    003504           1365#  12587   12617
TAB25    003514           1367#  12625   12645
TAB26    003524           1369#  12624   12646
TAB27    003534           1371#  12638   12656
TAB28    003544           1373#  12664
TAB29    003554           1375#  12663
TAB29A   003564           1377#  12674
TAB3     003214           1288#  12089   12104   12108
TAB30    003574           1380#  12745
TAB31    003604           1382#  12744
TAB32    003614           1384#  12758
TAB33    003624           1386#  12759
TAB34    003634           1388#  12752
TAB4     003224           1292#  12356   12371
TAB40    003644           1390#  12785   12786   12805   12806
TAB41    003654           1392#  12837   12858   12887   12908
TAB42    003664           1394#  12888   12909
TAB43    003674           1396#  12928
TAB45    003704           1398#  12951   12957   12970   12974   13007
TAB46    003714           1400#  12961
TAB47    003724           1402#  13006   14774
TAB47A   003734           1404#
TAB48    003744           1406#  13017   13027
TAB49    003754           1408#  12985
TAB5     003234           1296#  12231
TAB5A    003244           1300#  12877
TAB6     003254           1304#  12090   12266   12342   12392   12707   12722   12796   12826   12847   12898   12952   13026
                         13037   13128   14781
```

```
TAB6A   003264          1308# 15123
TAB7    003274          1312# 12440   12836   12857
TAB8    003304          1316# 12441   12452   12453
TAB9    003314          1320# 12446
TAPAB0  104174         16998  17013#
TA114   020760          5147   5152#
TA116   022434          5520   5529    5555#   5570
TBITVE= 000014           515#
TB114   020770          5153   5156#
TC114   021000          5157   5160#
TD114   021010          5161   5164#
TEMP    002710          1183# 19871   19892   19898   19906   19941   19948#  19997#  20051#  20055   20060   20069#  20073
                       20078  20087   20096   20100   20108   20156#  20166   20167   20176   20179   20475   20476   20488
                       20507  20511   20516   20533   20537   20542   21526   21537
TE102   017550          4830#
TE103   017572          4840#
TE104   017614          4850#
TE105   017636          4860#
TE106   017660          4870#
TE107   017702          4880#
TE110   017724          4890#
TE111   017746          4900#
TE112   017770          4910#
TE113   020212          4975#
TE113A  020432          5039#
TE114   020604          5113#
TE115   021754          5416#
TE115A  022164          5420    5437    5454    5480#
TE115B  022174          5421    5438    5455    5484#
TE115C  022202          5422    5439    5487#
TE115D  022210          5456    5490#
TE115F  022216          5476    5493#
TE116   022220          5495#
TE116A  022474          5516    5525    5573#
TE116B  022500          5500    5526    5577#
TE116C  022512          5509    5517    5582#
TE116D  022522          5499    5508    5518    5527    5586#
TE117   022532          5593#
TE117A  022644          5596    5623#
TE120   022650          5628#
TE120A  022724          5634    5644#
TE121   023000          5664#
TE122   023366          5752#
TE123   023640          5828#
TE124   024060          5893#
TE125   024312          5963#
TE125A  024552          5966    5998    6036#
TE126   025000          6127#
TE126A  025046          6134    6140#
TE126B  025522          6229    6268#
TE127   026040          6390#
TE127A  026240          6420    6452#
TE130   026502          6554#
TE130A  026760          6601    6638#
TF114   021020          5165    5168#
TG114   021030          5169    5172#
```

```
THRBIT  111704          18425   18474#
TH114   021040           5173    5176#
TIMDEL  114076          19060   19085   19100#
TIMEOU  052552          10632   10684#
TIMOUT  002750           1185#
TKVEC = 000060            522#
TM16A   050326          10146   10203#
TM16B   050206          10144   10175#
TM16C   050236          10150   10183#
TM16D   050266          10156   10191#
TM16E   050316          10137   10199#  10211   10215
TM16F   050320          10200#  10238
TM16A   050712          10172   10202   10267   10291   10294#
TOUT    132334           1479   16152   20155   21554#
TP    = ****** U         7235
TPA   = ****** U         7292
TPO   = ****** U         7260
TPVEC = 000064            523#
TRAPVE= 000034            521#   1448*   1449*
TRPFLG  132646          10543*  10567*  10594*  10627*  10695*  10724*  10777*  10839*  10901*  10932*  10952*  10991*  11036*
                        11092*  11142*  11195*  11246*  11302*  11349*  11388*  11439*  11479*  11580*  11674*  11768*  11811*
                        11891*  21640#
TRPO  = ****** U         6972
TRPOA   030354           6978    6984#
TRPOB   030362           6980    6987#
TRTVEC= 000014            516#
TRY   = ****** U         6836
TRYM  = ****** U         6870
TRYMA   027720           6876    6882#
TRYMB   027752           6886    6891#
TRYMC   030000           6894    6899#
TSD31   057140         11773#  11782
TSEND   111244          18306   18310#
TSFP1   052064         10541#
TSFP10  053246         10837#
TSFP11  053466         10899#
TSFP12  053562         10930#
TSFP13  053622         10948#
TSFP14  053746         10987#
TSFP15  054100         11032#
TSFP16  054266         11088#
TSFP17  054432         11138#
TSFP2   052150         10565#
TSFP20  054614         11193#
TSFP21  054766         11244#
TSFP22  055156         11300#
TSFP23  055322         11347#
TSFP24  055442         11386#
TSFP25  055616         11437#
TSFP26  055734         11477#
TSFP27  056276         11578#
TSFP3   052242         10592#
TSFP30  056604         11672#
TSFP31  057114         11766#
TSFP32  057234         11809#
TSFP33  057506         11889#
```

```
TSFP4   052336          10625#
TSFP5   052562          10693#
TSFP6   052652          10722#
TSFP7   053026          10775#
TSF10   053432          10843   10881#
TSF13   053742          10951   10981#
TSF14   054072          10990   11021#
TSF15   054260          11035   11077#
TSF16   054424          11091   11127#
TSF17   054606          11141   11182#
TSF2    052216          10569   10571   10573   10575   10580#
TSF20   054760          11197   11233#
TSF21   055150          11248   11289#
TSF22   055314          11304   11336#
TSF23   055434          11351   11376#
TSF24   055610          11390   11426#
TSF31   057226          11770   11798#
TSF6    053022          10729   10770#
TSF7    053212          10781   10819#
TSGPR0= ****** U         1602
TSGPR1= ****** U         1626
TSGPR2= ****** U         1650
TSGPR3= ****** U         1674
TSGPR4= ****** U         1698
TSGPR5= ****** U         1722
TSGPR6= ****** U         1746
TSLOOP  110524          18226#  18302   18307
TSMA    042710           9109    9112#
TSMB    042730           9114    9117#
TSMC    042740           9118    9121#
TSMMU0  014646           3963#
TSMMU1  034744           7983#
TSMMU2  035030           8001#
TSMMU3  036230           8255#
TSMMU4  036530           8328#
TSMMU5  037460           8514#
TSMMU6  037612           8545#
TSMMU7  042316           9019#
TSMMU8  042744           9124#
TSMMU9  043146           9165#
TSMM10  044046           9375#
TSMM11  044430           9457#
TSMM12  044666           9506#
TSMM13  045254           9583#
TSMM14  046322           9801#
TSMM15  047446          10036#
TSMM16  047766          10133#
TSMM6A  037720           8566#
TSMM6B  040356           8668#
TSMM6C  041116           8795#
TSMM6D  041616           8907#
TSM16A  050062          10150#  10182
TSM16B  050110          10156#  10190
TSM16C  050134          10162#  10198
TSM16D  050200          10170#  10223
TSM7    042676           9052    9064    9108#
```

```
TSM9    043302'          9178    9188    9206#   9246
TSPSWB- ****** U         1771
TSTADD  002772           1231#
TSTLOC  003122           1260#   10743   10745   10787   10790   10799   10803   10849   10852   10861   10865   10908   10956
                        10959   10962   10995   10998   11001   11005   11040   11043*  11044   11047   11051   11052   11096
                        11099   11102   11146   11149*  11150   11153   11157   11201   11204   11207   11211   11252   11255*
                        11256   11259   11263   11270   11308   11311*  11312*  11314   11355   11358*  11361   11394   11397*
                        11398*  11400   11407   11443   11446   11449   11453   11584   11588   11591   11601   11604   11608
                        11615   11618   11622   11678   11682   11685   11695   11698   11702   11709   11712   11716   11815
                        11819*  11821   11828*  11830   11837*  11839   11895   11899*  11901   11908*  11910   11917*  11919
                        12045*  12046   12047*  12048   12058   12062   12130   12131*  12141   12197*  12200   14833   14839
                        14871   14877   14909   14910*  14911*  14912   14913*  14917   14921   14949   14950*  14955   14987
                        14988*  14989*  14990   14991*  14995   14999   15027   15028*  15029   15033   15037   15041   15065
                        15066*  15067*  15068   15069*  15073   15077   15081   16293*  16295*  16298   16336   16351*  16388*
                        16390*  16392   16397   16447   16448   16455   16460*  16464   16584   16585*  16586   16591*  16592*
                        16598   16601*  16602*  16607   16667*  16671*  16674   16681   16684   16690   16812   16813   16815
                        16820*  16825   16830*  16914*  16916*  16919   16933*  16935   17001*  17003   17070*  17073   17148*
                        17151   17203*  17205   17219*  17221   17326*  17328   17334   17341   17343*  17345   17352   17354*
                        17356   17367   17369*  17371   17377   17379*  17381   17522   17525   17536   17545*  21200
TSTLUP  107412           17976#  18058   18063
TST1    004534           1513#
TST10   103024           16612   16665#
TST11   103234           16742#
TST12   103442           16780   16811#
TST13   103610           16907#
TST14   104072           16951   16996#
TST15   104204           17010   17057#
TST16   104400           17080   17136#
TST17   104546           17156   17192#
TST2    101530           16151#
TST20   104740           17318#
TST21   105422           17390   17460#
TST22   105600           17482   17521#
TST23   105762           17551   17610#
TST24   106266           17723#
TST25   107004           17795   17811#
TST26   107114           17923#
TST27   110174           18167#
TST3    101656           16174   16203#
TST30   111262           18343#
TST31   111434           18382   18419#
TST32   111712           18472   18516#
TST33   112216           18600#
TST34   112350           18660#
TST35   112520           18664   18693   18743#
TST36   113004           18747   18827#
TST37   113152           18831   18898#
TST4    101724           16289#
TST40   113446           18902   18978#
TST41   113624           18982   18986   19007   19046#
TST42   114106           19050   19098   19117#
TST43   114146           19125   19158#
TST44   114244           19198#
TST45   114310           19208   19238#
TST46   114506           19247   19295#
TST47   114550           19299   19306   19339#
```

```
TST5    102242          16352   16385#
TST50   115124          19450#
TST51   115354          19476   19539#
TST52   115644          19544   19546   19572   19578   19625#
TST53   116044          19647   19676#
TST54   116352          19708   19768#
TST55   116712          19770   19772   19774   19830   19862#
TST56   117154          19933#
TST57   117274          19978#
TST6    102374          16403   16445#
TST60   117456          19983   20038#
TST61   120366          20216#
TST62   120644          20218   20220   20298#
TST63   121030          20300   20302   20352#
TST64   121130          20354   20356   20390#
TST65   121312          20392   20462#
TST66   122012          20549#
TST7    102602          16473   16579#
TS10    044346           9391    9402    9413    9427    9435#
TS10D1  053436          10844   10856   10885#
TS10D2  053446          10846   10853   10889#
TS10D4  053456          10864   10893#
TS11    044546           9465    9470    9475    9481#
TS11D1  053552          10904   10924#
TS12    045204           9524    9531    9538    9545    9552    9559    9566#
TS14    046726           9853    9875    9886#
TS15    047612          10043   10051   10060   10085#
TS16    050616          10261   10268#
TS16A   050610          10265#  10280
TS1822  045660           9701#
TS26D0  056214          11482   11488   11511   11517   11547#
TS26D1  056224          11491   11497   11520   11526   11551#
TS26D2  056234          11500   11506   11529   11555#
TS26D3  056244          11489   11518   11559#
TS26D4  056254          11498   11527   11563#
TS26D5  056264          11507   11567#
TS27D0  056522          11585   11641#
TS27D1  056532          11645#
TS27D2  056542          11649#
TS27D3  056552          11595   11653#
TS27D4  056562          11609   11657#
TS27D5  056572          11623   11661#
TS30D0  057032          11679   11735#
TS30D1  057042          11739#
TS30D2  057052          11743#
TS30D3  057062          11689   11747#
TS30D4  057072          11703   11751#
TS30D5  057102          11717   11755#
TS32D0  057424          11816   11858#
TS32D1  057434          11862#
TS32D3  057454          11822   11870#
TS32D4  057464          11831   11874#
TS32D5  057474          11840   11878#
TS33D0  057700          11896   11938#
TS33D1  057710          11942#
TS33D2  057720          11946#
```

```
TS33D3  057730          11902   11950#
TS33D4  057740          11911   11954#
TS33D5  057750          11920   11958#
TS6DA   053002          10736   10749   10761#
TS6DAT  053012          10730   10732   10765#
TS7     042566           9032    9039    9080#
TS7DA1  053216          10782   10794   10823#
TS7DA2  053226          10784   10791   10827#
TS7DA4  053236          10802   10831#
TS7FIN  042742           9076    9123#
TS9FIN  044044           9202    9374#
TT    - ****** U         7088    7115
TTR   - ****** U         7142
TYPDS - 104405          19821   19827   21705   22396#
TYPE  - 104401           1494   19685   19686   19703   19822   19828   21247   21264   21266   21269   21271   21275   21294
                        21492   21703   21706   21813   21821   21985   22098   22173   22203   22204   22207   22220   22231
                        22250   22301   22304   22308   22392#  22438
TYPOC - 104402          21255   21291   22206   22393#
TYPON - 104404          22395#
TYPOS - 104403          22394#
T10FIN  044430           9430    9456#
T11FIN  044666           9477    9505#
T114    020736           5128    5138    5146#
T116    022402           5502    5511    5541#   5552
T12FIN  045252           9562    9582#
T122A   023464           5758    5769    5779#
T122B   023616           5756    5818#
T123A   024022           5833    5876#
T123B   024032           5880#
T123C   024040           5883#
T123D   024046           5837    5886#
T123E   024050           5869    5888#
T123F   024054           5873    5890#
T124A   024254           5898    5946#
T124B   024264           5950#
T124C   024272           5953#
T124D   024300           5902    5956#
T124E   024302           5939    5958#
T124F   024306           5943    5960#
T13FIN  045660           9630    9700#
T14     046752           9871    9891#
T14FIN  047324           9711    9880    9913   10003#
T15     047662          10046   10054   10063   10102#
T15A    047740          10068   10072   10076   10121#
T15FIN  047766          10080   10131#
UDPAR0- 177660            318#
UDPAR1- 177662            319#
UDPAR2- 177664            320#
UDPAR3- 177666            321#
UDPAR4- 177670            322#
UDPAR5- 177672            323#
UDPAR6- 177674            324#
UDPAR7- 177676            325#
UDPDR0- 177620            296#   8582*   8810*
UDPDR1- 177622            297#
UDPDR2- 177624            298#
```

```
UDPDR3- 177626            2990
UDPDR4- 177630            3000
UDPDR5- 177632            3010
UDPDR6- 177634            3020
UDPDR7- 177636            3030
UFDFLG  004056           1418*   1429#  21852
UFDSET- 000001            525#   1412   21802   21847
UIPAR0- 177640            3070
UIPAR1- 177642            3080
UIPAR2- 177644            3090
UIPAR3- 177646            3100
UIPAR4- 177650            3110
UIPAR5- 177652            3120
UIPAR6- 177654            3130
UIPAR7- 177656            3140
UIPDR0- 177600            2850   8682*   8922*
UIPDR1- 177602            2860
UIPDR2- 177604            2870
UIPDR3- 177606            2880
UIPDR4- 177610            2890
UIPDR5- 177612            2900
UIPDR6- 177614            2910
UIPDR7- 177616            2920
UNXPIR  034242            7737    7804#
UQUIET  004060           1421*   1430#  21802   21858
UVAD  - ****** U         12772
VIR1    045542            9598    9660#
VIR2    047154            9850    9948#
VIR3    047272            9830    9852    9987#
VMKOR   004062            1415    1417    1426    1431#
VQBE1   002656           1165#  20225*  20254*  20303*  20399*
VQBE2   002676           1175#  20401*
VQPR1   002660           1166#  20226*  20255*  20304*  20400*
VQPR2   002700           1176#  20402*
WC      002652           1163#  20101*  20135*  20477*  21527*  21541*  21546*
WC2     002672           1173#  20478*
WLDTRP  132636          12835   12886   216339
XBUF  - 177566           1203#  19175   19206*  19260*  19390*  19483*  19585*  19629*  19634*  19653*
XCBIT   016500           4491#
XCSR  - 177564           1204#  19200   19203   19207   19240   19242*  19243   19245*  19267*  19273*  19277*  19279*  19300*
                        19305   19348*  19349   19357   19368*  19382*  19388   19395*  19400*  19416*  19458*  19471*  19474*
                        19482*  19488*  19502*  19550*  19553*  19554   19562*  19570*  19573*  19574   19576*  19583   19591*
                        19593*  19626*  19627   19632   19641*  19645*  19651   19658*  19660*
XDF1  - ****** U         12467
XME100  017504            4808#
XME101  017526            4819#
XOR   - ****** U          4699
XRCC  - ****** U          4613
$APTHD  000232             561     567#
$ASTAT- ****** U         21919   21934
$ATYC   133742           21890   21892#
$ATY1   133716           218888
$ATY3   133724           21889#  21970
$ATY4   133734           21826   21891#
$AUTOB  001134             598#   1504*  22200   22323
$BASE   001254             664#
```

```
$BDADR  001122          593#  16159*  16165*  16171*  16355*  16615*  16954*  17013*  17086*  17161*  17627*  17628*  17652*
                      17653*  17761*  17762*  17955*  17956*  18198*  18199*  21198   21212   21214   21216   21230   21232
                      21421*  21554*
$BDDAT  001126          595#  17532*  17543*  17552*  19173*  19269*  19270   19567*  19568   19588*  19589   19637*  19639
                      19643   21210   21226   21230
$BELL   001170          615#  19703   21813   21846
$CDW1   001260          666#
$CDW2   001262          667#
$CHARC  134514        21987*  21997*  22004   22030*  22035#
$CKSWR  135172        22192#  22400
$CMTAG  001100          581#   1437    1438    1446    1452    1453    1454
$CM3  = 000000          611#
$CM4  = 000002          611#    612#    613#
$CNTLG  135727        22203   22318#
$CNTLU  135722        22220   22317#
$CPUOP  001226          638#
$CRLF   001175          617#  19685   21247   21266   21271   21275   21821   21846   21986   22038   22231   22317
$DBLK   135162        22139   22173   22181#
$DDW0   001264          668#
$DDW1   001266          669#
$DDW10  001310          678#
$DDW11  001312          679#
$DDW12  001314          680#
$DDW13  001316          681#
$DDW14  001320          682#
$DDW15  001322          683#
$DDW2   001270          670#
$DDW3   001272          671#
$DDW4   001274          672#
$DDW5   001276          673#
$DDW6   001300          674#
$DDW7   001302          675#
$DDW8   001304          676#
$DDW9   001306          677#
$DEVCT  001210          629#
$DEVM   001256          665#
$DOAGN  133030        21699   21708   21714#
$DTBL   135152        22142   22177#
$ENDAD  133020          548    1492   21710#  21841   21881
$ENDCT  132766         1452   21701#
$ENDMG  133037        21703   21718#
$ENULL  133034        21706   21717#
$ENV    001220          634#   1498    5666   18461   18979   19296   19340   19682   19773   21823   21897   21921   21965
$ENVM   001221          635#   1475   21899   21967   21972
$EOP    132724        19981   20041   20044   20395   20465   20468   20551   21690#
$EOPCT  132760         1452*  21698#  21702
$ERFLG  001103          584#  21726   21757   21759   21765*  21787   21808*  21846
$ERMAX  001115          590#   1455*  21759   21782*  21787
$ERROR  133342         1446    7388    7415   21801#
$ERRPC  001116          591#  21193   21194   21196   21198   21200   21202   21204   21206   21208   21210   21212   21214
                      21216   21218   21220   21222   21224   21226   21228   21230   21253   21815*  21816*  21817   21846
$ERRTB  001324          702#  21261
$ERTTL  001112          588#  21814*  21846
$ESCAP  001166          614#   1454*  21781*  21837   21839   21846
$ETABL  001220          633#
$ETEND  001324          573     686#
```

```
$FATAL  001202          626# 21925*
$FFLG   134162        21888* 21891* 21919   21928* 21936#
$FILLC  001156          609# 21990  22038
$FILLS  001155          608# 22038
$GDADR  001120          592#
$GDDAT  001124          594# 16297* 16467* 16668* 17092* 17170* 17331* 17348* 17359* 17374* 17384* 18356* 18357*
                      18368* 18369* 18379* 18426* 18429* 18434* 18445* 18453* 18459* 18784* 18798* 18839* 18844*
                      18910* 18915* 18937* 19272* 19638* 19911* 21196  21200  21202  21208  21218  21220  21222
                      21224  21226
$GET42  133010        21692  21707#
$GTSWR  135242        22204# 22398
$HD   - 000001          256    257
$HIBTS  000232          568#
$HIOCT  136056        22355* 22360#
$ICNT   001104          585# 21772* 21773  21775* 21786
$ILLUP  136302        22408  22424  22441#
$INTAG  001135          599# 22232  22323
$ITEMB  001114          589# 21250  21817* 21825  21846
$LF     001176          618# 21846  22038  22308  22317
$LFLG   134161        21929* 21935#
$LPADR  001106          586#  1456* 21763* 21779* 21784  21786
$LPERR  001110          587#  1457* 21763  21780* 21786  21836
$MADR1  001232          651#
$MADR2  001236          655#
$MADR3  001242          658#
$MADR4  001246          661#
$MAIL   001200          569    573    624#  1474   1498   21778  21823  21965
$MAMS1  001230          645#
$MAMS2  001234          653#
$MAMS3  001240          656#
$MAMS4  001244          659#
$MBADR  000234          569#
$MFLG   134160        21889* 21895  21930* 21934#
$MNEW   135745        22207  22321#
$MSGAD  001214          631# 21905* 21908
$MSGLG  001216          632# 21910*
$MSGTY  001200          625# 21903  21911* 21923  21927*
$MSWR   135734        22204  22319#
$MTYP1  001231          646#
$MTYP2  001235          654#
$MTYP3  001241          657#
$MTYP4  001245          660#
$MXCNT  133340        21776  21786#
$NULL   001154          607# 21992  22038
$NWTST- 000000         1512# 16150# 16202# 16288# 16384# 16444# 16578# 16664# 16741# 16810# 16906# 16995# 17056#
                      17135# 17191# 17317# 17459# 17520# 17609# 17722# 17810# 17922# 18166# 18342# 18418# 18515#
                      18599# 18659# 18742# 18826# 18897# 18977# 19045# 19116# 19157# 19197# 19237# 19294# 19338#
                      19449# 19538# 19624# 19675# 19767# 19861# 19932# 19977# 20037# 20215# 20297# 20351# 20389#
                      20461# 20548#
$OCNT   134742        22070* 22099* 22112#
$OMODE  134744        22065* 22069* 22074  22077* 22088* 22114#
$OVER   133324        21738  21756  21764  21774  21783#
$PASS   001206          628#  1474* 18463  18981  19298  19342  19545  19680  19771  21695* 21696* 21704  21717
                      21770  21787
$PASTM  000240          571#
$POWER  136310        22439  22444#
```

F5


```
$PWRDN  136142          1450   22408#  22436
$PWRMG  136276         22439#
$PWRUP  136214         22418   22424#
$QUES   001174           616#  21846   22038   22250   22301   22317
$RDCHR  135454         22263#  22401
$RDDEC- ****** U       22404
$RDLIN  135604         22293#  22402
$RDOCT  135756         22333#  22403
$RDSZ - 000010         22286#
$RTNAD  133032         21716#
$R2A  - ****** U       22404
$SAVRE- ****** U       22404
$SAVR6  136306         22417*  22425   22426*  22427*  22443#
$SCOPE  133054          1444   21735#
$SETUP- 000137          1435#   1443    1444    1446    1448    1450    1452    1453    1454    1456    1492    1495   21693
                       21736   21807   21833   21841   22187   22323
$STUP - 177777          1435#
$SVLAD  133270         21746   21777#
$SVPC - 000232          546#    551
$SWR  - 167400          244#    256     261     262     263     264     265     266     267     613     614     615    1453
                       1454    1456    1457    1514   16152   16204   16290   16386   16446   16580   16666   16743   16812
                      16908   16997   17058   17137   17193   17319   17461   17522   17611   17724   17812   17924   18168
                      18344   18420   18517   18601   18661   18744   18828   18899   18979   19047   19118   19159   19199
                      19239   19296   19340   19451   19540   19626   19677   19769   19863   19934   19979   20039   20217
                      20299   20353   20391   20463   20550   21687   21694   21709   21715   21717   21727   21728   21729
                      21730   21731   21737   21749   21751   21752   21757   21758   21759   21766   21767   21768   21780
                      21783   21786   21793   21794   21795   21796   21797   21811   21818   21830   21834   21846   22440
$SWREG  001222          636#   1477
$SWRMK- 000300          245#    267     268   21731   21732   21753   21754
$TESTN  001204          627#   9709*  21778*
$TIMES  001164          613#   1453*  21694*  21766*  21773   21776*  21786
$TKB    001146          604#  22011   22018   22038   22185   22196   22213   22267   22273
$TKS    001144          603#  22009   22016   22038   22185   22194   22210   22234*  22265   22271
$TMPO   001160          536*    539*    611#  17630*  17631   17655*  17656   17724*  17793   17931*  17961   18175*  18204
                      18525*  18530*  18542   18554*  18559*  18576   18604*  18613*  18623   18668*  18676   19994*  20014
                      20234*  20236*  20237*  20247   20263*  20265*  20266*  20276   20357*  20368   21281*  21282*  21283*
                      21284*  21285   21450*  21463   21479*  21480
$TMP1   001162          612#  16447*  16460   18622*  21193   21194   21196   21198   21200   21202   21204   21206   21208
                      21210   21212   21214   21216   21218   21220   21222   21224   21226   21228   21230   21232   21245*
                      21246*
$TN   - 000067          256#    1512    1514#   16150  16152#   16174   16202  16204#   16288  16290#   16352   16384  16386#
                      16403   16444  16446#   16473   16578  16580#   16612   16664  16666#   16741  16743#   16780   16810
                      16812#  16906   16908#   16951   16995  16997#   17010   17056  17058#   17080   17135  17137#   17156
                      17191   17193#  17317   17319#   17390   17459  17461#   17482   17520  17522#   17551   17609  17611#
                      17722   17724#  17795   17810   17812#   17922   17924#   18166  18168#   18342  18344#   18382   18418
                      18420#  18472   18515   18517#   18599   18601#   18659  18661#   18664   18693   18742  18744#   18747
                      18826   18828#  18831   18897   18899#   18902   18977  18979#   18982   18986   19007   19045  19047#
                      19050   19098   19116   19118#   19125   19157   19159#   19197  19199#   19208   19237  19239#   19247
                      19294   19296#  19299   19306   19338   19340#   19449   19451#   19476   19538  19540#   19544   19546
                      19572   19578   19624   19626#   19647   19675   19677#   19708   19767  19769#   19770   19772   19774
                      19830   19861   19863#   19932   19934#   19977   19979#   19983   20037  20039#   20215  20217#   20218
                      20220   20297   20299#   20300   20302   20351   20353#   20354   20356   20389  20391#   20392   20461
                      20463#  20548   20550#
$TPB    001152          606#  22027*  22038
$TPFLG  001157          610#  21959   22038
$TPS    001150          605#  22025   22038
```

CROSS REFERENCE TABLE USER SYMBOLS SEQ 0472

```
$TRAP   136060           1448   22369#
$TRAP2  136102         22380#   22391
$TRP  = 000013         22384#   22393#  22394#  22395#  22396#  22397#  22398   22399#  22400   22401#  22402#  22403#  22404#
$TRPAD  136114         22374    22391#
$TSTM   000236           570#
$TSTNM  001102           583#   21246   21693*  21726   21755   21777*  21778   21783   21787   21810   21846
$TTYIN  135712         22294    22295   22312   22316#
$TYPBN= ****** U       22397
$TYPDS  134746         22127#   22396
$TYPE   134164         21916    21959#  22384   22392
$TYPEC  134376         21989    21996   22003   22008#  22236
$TYPEX  134516         22031    22033   22036#
$TYPOC  134544         22068#   22393
$TYPON  134560         22067    22070#  22395
$TYPOS  134520         22063#   22394
$UNIT   001212           630#
$UNITM  000242           572#
$USWR   001224           637#
$VECT1  001250           662#
$VECT2  001252           663#
$XOFF = 000023         22013    22038
$XON  = 000021         22020    22038   22278
$XTSTR  133066         21740#
$$GET4= 000000         21709#
$OFILL  134743         22064*   22068*  22078   22113#
$40CAT= ****** U       21737    21820
.     = 136320           528#     532#    535#    538#    546     547#    549#    551#    557     558#    560#    562#    580#
                         619     1182#   1184#   1239#   1240#   1244#   1245#   1247#   1248#   1249#   1261#   1441    1456
                        1457     1508#  17064   17142   17193   17813   18387#  19100#  19987   21297#  21502#  21717   21721
                       21786    21787   21846   21937#  22038   22181#  22185   22316#  22317   22323   22420   22442
.$ASTA= ****** U       21889    21892
.$X   = 000232           557#     562

. ABS.  136320     000

ERRORS DETECTED:  0

COKDAA,COKDAA/SOL/NL:TOC/CRF:SYM=ORION.SML/ML,COKDAA.P11
RUN-TIME: 112 51 5 SECONDS
RUN-TIME RATIO: 837/170=4.9
CORE USED:  37K  (74 PAGES)
```