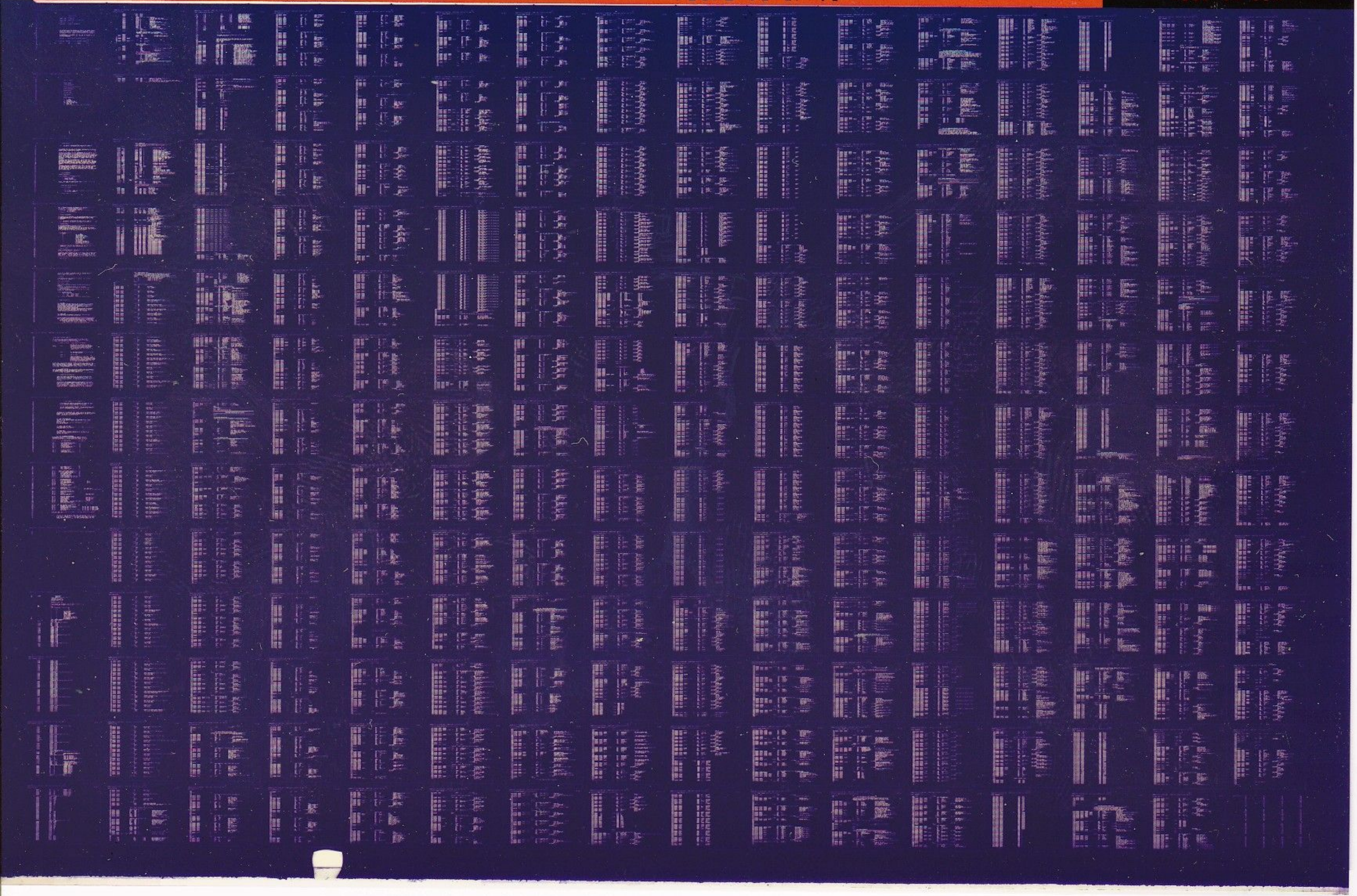
KDJ11-B KDJ11-B CLUSTER DIAG
COKDACO

COPYRIGHT (c) 1984
AH-T854C-MC
FICHE 01 OF 03

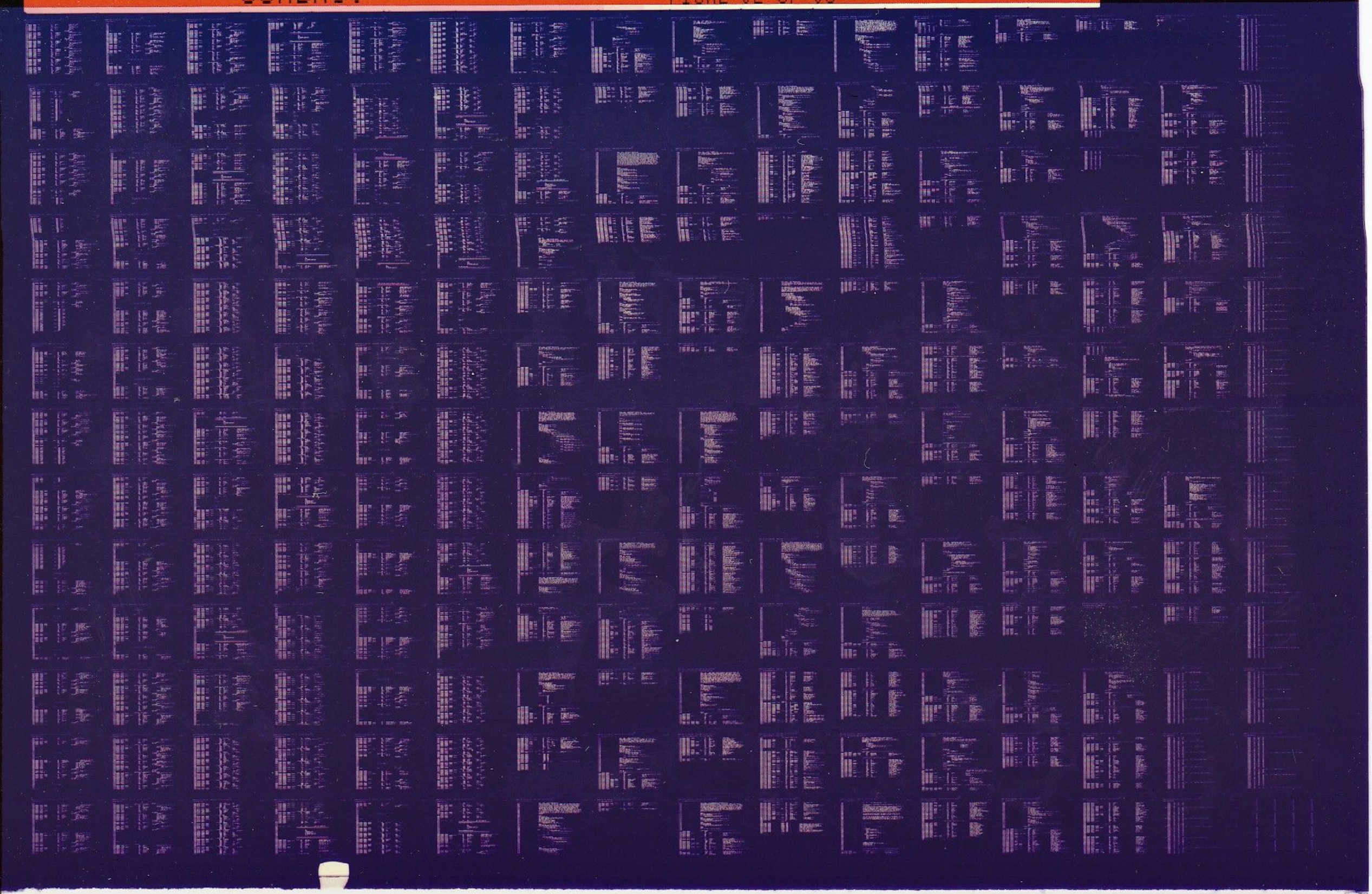
FEB 1985
digital
Made In USA



KDJ11-B KDJ11-B CLUSTER DIAG
COKDACO

COPYRIGHT (c) 1984
AH-T854C-MC
FICHE 02 OF 03

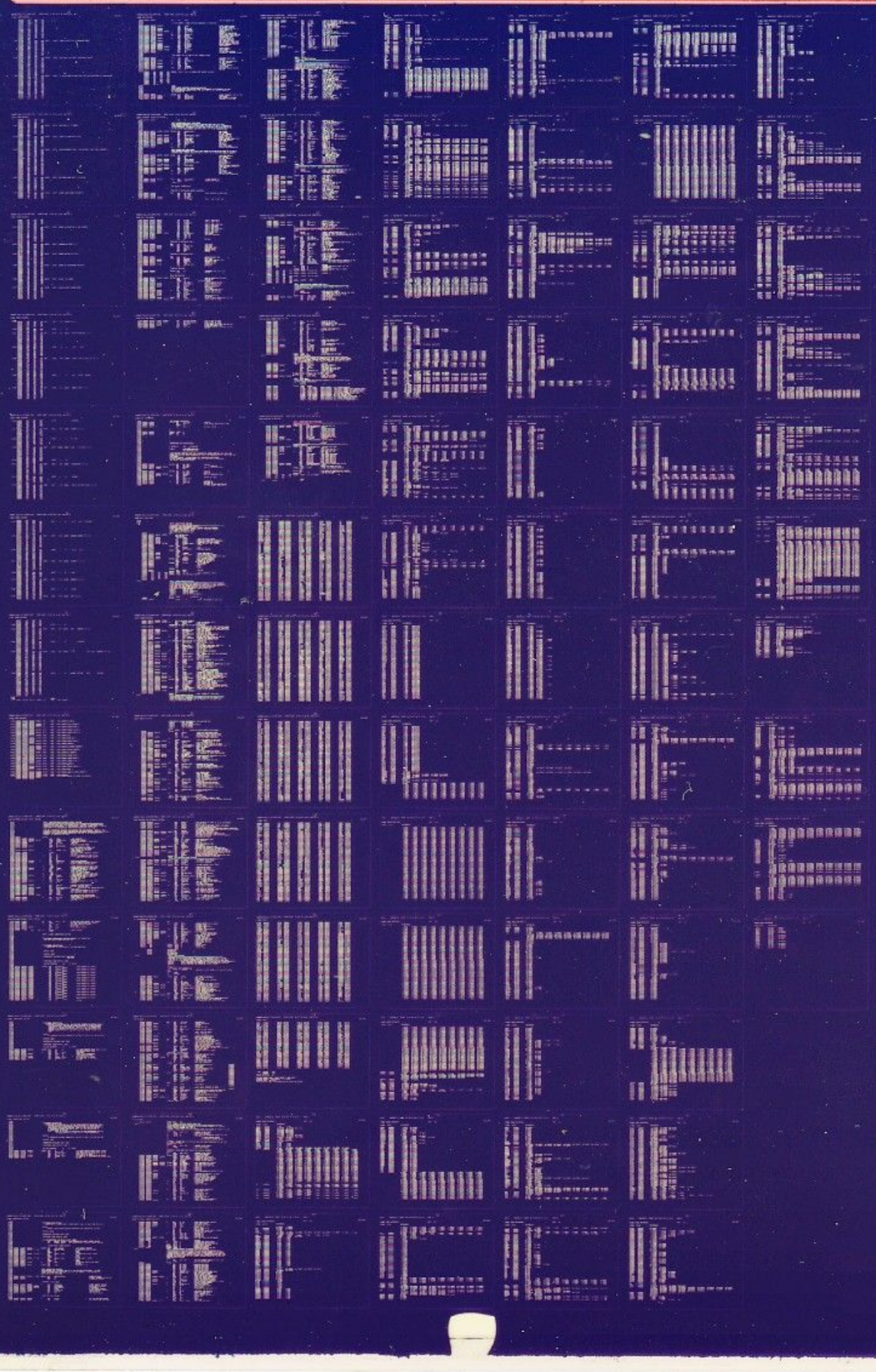
FEB 1985
digital
Made In USA



KDJ11-B KDJ11-B CLUSTER DIAG
COKDACO

COPYRIGHT (c) 1984
AH-T854C-MC
FICHE 03 OF 03

FEB 1985
digital
Made In USA



```
                                                    .REM &

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

                    PRODUCT CODE:  AC-T853C-MC

                    PRODUCT NAME:  COKDACO KDJ11-B CLUSTER DIAG.

                    PRODUCT DATE:  JULY, 1984

                    MAINTAINER:    DIAGNOSTIC ENGINEERING




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

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

                    COPYRIGHT (C) 1984 BY DIGITAL EQUIPMENT CORPORATION


            THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

                    DIGITAL         PDP             UNIBUS          MASSBUS
                    DEC             DECUS           DECTAPE
```

```
44
45
46
47                                      TABLE OF CONTENTS
48
49
50                                  1.      PROGRAM ABSRACT
51
52                                  2.      SYSTEM REQUIREMENTS
53
54                                  3.      LOADING AND STARTING PROCEDURES
55
56                                  4.      SPECIAL ENVIRONMENTS
57
58                                  5.      PROGRAM OPTIONS
59
60                                  6.      EXECUTION TIMES
61
62                                  7.      ERROR INFORMATION
63
64                                  8.      EXAMPLES
65
66                                  9.      PROGRAM DESCRIPTION
67
68                                          9.1  J11 CODE
69                                          9.2  CACHE CODE
70                                          9.3  ON-BOARD ROM CODE
71                                          9.4  LINE TIME CLOCK CODE
72                                          9.5  SERIAL LINE UNIT CODE
73                                          9.6  Q22BE CODE
74                                          9.7  LIST OF SUBTESTS SHOWING CACHE/APT DEPENDENCY
```

```
                                1.   PROGRAM ABSTRACT

                                  ****   SEE SPECIAL INSTRUCTIONS FOR CACHE TESTING UNDER APT,     ****
                                  ****   SECTION 9.2, AND EEROM TESTING UNDER APT, SECTION 9.3     ****


                                THIS PROGRAM TESTS OUT KDJ11-B CPU BOARD, INCLUDING THE J11 CHIP SET,
                                ON-BOARD CACHE, ON-BOARD ROM'S, INCLUDING 16 BIT AND THE 8 BIT EEROM,
                                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.
```

```
133                     NOTE: IF TRYING TO RESTART THE PROGRAM IN AN ARBITRARY PLACE AFTER
134                     HALT ON BREAK THE FOLLOWING REGISTERS SHOULD BE SET UP:
135                     17777572=0      TO DISABLE MEMORY MANAGEMENT
136                     17777520=1000   TO CLEAR DIAGNOSTIC MODE (BIT 8), BUT STILL SAVE
137                                     HALT ON BREAK
138                     17777746=400    TO FLUSH THE CACHE
139
140
141                     4.  SPECIAL ENVIRONMENTS
142
143                     THE PROGRAM IS APT COMPATIBLE. IT CAN ALSO BE RUN UNDER THE UFD MONITOR.
144                     IN THOSE CASES NONE OF THE STANDARD ERROR PRINTOUTS OCCUR. REFER
145                     TO CORRESPONDING DOCUMENTS ON RUNNING PROCEDURES IN APT AND UNDER
146                     UFD MONITOR.
147
148                       ****   SEE SPECIAL INSTRUCTIONS FOR CACHE TESTING UNDER APT,     ****
149                       ****   SECTION 9.2, AND EEROM TESTING UNDER APT, SECTION 9.3     ****
150
151                     5.  PROGRAM OPTIONS
152
153                     THE Q22 BUS EXERCISER IS UTILIZED IF IT IS PRESENT IN THE SYSTEM AND
154                     THE DIAGNOSTIC IS NOT RUNNING IN UFD MODE.
155                     STANDARD CAPABILITIES OF LOOPING ON TEST AND ON ERROR ARE PROVIDED.
156                     IN ORDER TO RUN THE EXTENSIVE CACHE DATA RAM TEST BIT 7 HAS TO BE SET
157                     IN THE SOFTWARE SWITCH REGISTER. TO RUN THE EXTENSIVE CACHE TAG RAM TEST
158                     BIT 6 HAS TO BE SET IN THE SOFTWARE SWITCH REGISTER.
159
160
161                     SWITCH REGISTER SELECTION:
162                         BIT NUMBER                          USE
163                             15                      HALT ON ERROR
164                             14                      LOOP ON PRESENT TEST
165                             13                      INHIBIT ERROR TYPEOUTS
166                     ***     12                      EEROM SUBTEST RUN SWITCH
167                             11                      INHIBIT ITERATIONS
168                             10                      BELL ON ERROR
169                              9                      LOOP ON ERROR
170                              8                      LOOP ON TEST IN SWR<5-0>
171                              7                      DO EXTENSIVE DATA RAM TEST
172                              6                      DO EXTENSIVE TAG RAM TEST
173                            5-0                      SUBTEST NUMBER TO LOOP ON (BIT 8)
174
175                     *** RUNNING WITH THIS TEST WILL CHANGE ALL BUT THE FIRST 109 BYTES OF
176                         EEROM DATA!  BE SURE TO READ INSTRUCTIONS IN SECT. 9.3
177
178                     6. EXECUTION TIMES
179
180                     WITHOUT EXTENSIVE RAM TESTS, THE DIAGNOSTIC RUNS IN UNDER 15 SECONDS.
181                     WITH THEM, IT TAKES ABOUT 2 MINUTES.
182
183                     IN ADDITION, WHEN THE EEROM TEST BIT IS ON, PASS 1 TAKES AN ADDITIONAL
184                     AMOUNT OF TIME OF 1 MINUTE FOR 15MHZ J-11, TO ABOUT 40 SEC. FOR THE
185                     20MHZ. VERSION.  THESE TEST TIMES ARE FOR THE 2K EEROM, FOR 8K PARTS,
186                     MULTIPLY BY 4.  IF RUNNING UNDER APT, THE SYSTEM MANAGER MUST BE SURE
187                     THE FIRST PASS RUN TIME ALLOWS ADEQUATE TIME FOR COMPLETION OF THIS TEST!
188
189                     7. ERROR INFORMATION
```

```
190
191                     IN THE CASE OF ERRORS, A FAILING PC AND TEST NUMBERS ARE GIVEN.
192                     WHERE IT IS POSSIBLE, EXPECTED AND RECEIVED DATA ARE GIVEN.
193                     FOR AN EXAMPLE, SEE SECTION 8.
194
195                     8. EXAMPLES
196
197                     AFTER BOOTING XXDP+ AND STARTING THE PROGRAM, THE FOLLOWING
198                     WILL APPEAR ON THE TERMINAL:
199
200                     KDJ11-B CPU DIAGNOSTIC
201
202                     SWR = XXXXXX    NEW =
203
204                     WHERE XXXXXX CORRESPOND TO PRESENT SOFTWARE SWITCH REGISTER
205                     SETTING.
206                     AFTER "NEW" AN OPERATOR CAN DO ONE OF THE FOLLOWING:
207                     1) TYPE IN A NEW SOFTWARE SWITCH REGISTER SETTING FOLLOWED
208                     BY CARRIGE RETURN OR
209                     2) JUST TYPE IN CARRIAGE RETURN IN WHICH CASE THE SOFTWARE
210                     REGISTER WILL REMAIN UNCHANGED.
211
212                     EXAMPLE OF ERROR PRINTOUT:
213                             ERROR IN TAG STORE
214                              TEST   ERROR   ADDRESS ADDRESS
215                               #      PC     <21-16> <15-0>
216                              27    105620  66600   000000
217                     NOTE: THIS MAY NOT CORRESPOND TO THE ACTUAL PROGRAM COUNTER.
218
219
220
221                     9. PROGRAM DESCRIPTION
222
223
224                     9.1  J11 CODE
225                     THIS PORTION OF THE CODE TESTS OUT THE J11 CHIP SET. IT IS BROKEN
226                     INTO 3 PIECES: CPU TESTS, WHICH VERIFY DIFFERENT INSTRUCTIONS IN
227                     DIFFERENT MODES AND DIFFERENT TRAP CONDITIONS; MMU TESTS, WHICH
228                     VERIFY DIFFERENT FUNCTIONS OF MMU; AND FFP TESTS, WHICH DO
229                     DIFFERENT FLOATING POINT INSTRUCTIONS.
230                     THIS PORTION OF THE CODE HAVE BEEN WRITTEN IN CLOSE RELATIONSHIP
231                     WITH THE J11 MICROCODE. THEREFORE, EVEN THOUGH NOT ALL POSSIBLE
232                     INSTRUCTIONS IN ALL POSSIBLE ADDRESSING MODE HAVE BEEN TESTED, AN
233                     ATTEMPT HAS BEEN MADE TO EXERCISE ALL OF THE MICROCODE.
234
235
236                     9.2  CACHE CODE
237                     THIS PORTION OF THE DIAGNOSTIC VERIFIES ALL CACHE FUNCTIONS.
238                     DATA AND TAG RAM'S TESTS ARE ALSO INCLUDED.
239
240             ********************************************************************************
241                     NOTE: IN ORDER TO RUN EXTENSIVE CACHE RAM'S TESTS THE CORRESPONDING
242                     BITS IN THE SOFTWARE SWITCH REGISTER SHOULD BE SET. SEE SECTION 5.
243                     ----IN PARTICULAR, THE NUMBER IN $USWR (SWARE REG #2) SHOULD EQUAL
244                         THE "FIRST PASS RUN TIME" SET UP AT INSTALLATION---- (THE EXCEPTION
245                             IS WHEN $USWR IS SET TO ZERO, NO CONDITION APPLIES)
246             ********************************************************************************
```

```
247
248                          TESTING CACHE FUNCTIONS UNDER APT
249
250                          THE TESTING OF CACHE RELATED FUNCTIONS UNDER APT IS FACILITATED
251                          BY USE OF $USWR CONTAINING A NUMBER WHICH WILL INDICATE THE NUMBER
252                          OF PASSES WITH INCLUSIVE CACHE TESTS AS FOLLOWS:
253
254                                 (ALL NUMBERS IN DECIMAL, $USWR LOADED BY APT MANAGER )
255
256                                                        ONE PASS WITH ALL SUBTESTS FOLLOWED BY
257                          DEFAULT       $USWR = 0       CONTINUOUS TESTING OF NON-CACHE DEPENDENT
258                          SETTING                       SUBTESTS UNTIL APT BREAK IS ENCOUNTERED
259                                                        (APT SYSTEM MUST NOT BREAK IN FOR THIS
260                                                         "FIRST PASS RUN TIME", NOW ASSUMED TO BE
261                                                         APPROX 16 SEC. )
262
263
264                                       $USWR < 16       ONLY NON-CACHE DEPENDENT TESTS WILL BE RUN
265                                                         APT MAY BREAK ANYTIME
266
267                                       $USWR = 16       SAME AS $USWR = 0
268
269                                       $USWR > 16       $USWR IS DIVIDED BY 16 (PASS TIME ) TO GET A
270                                                        NUMBER OF PASSES WHICH  INCLUDE ALL SUBTESTS
271                                                        (INCLUDING CACHE)  AFTER THIS NUMBER OF PASSES,
272                                                        CONTINUOUS TESTING WITH ONLY NON-CACHE DEPEND-
273                                                        ENT SUB-TESTS IS CONTINUED DURING WHICH TIME
274                                                        APT MAY BREAK WITH NO BAD EFFECTS.
275
276                          IN THIS CASE, THE APT SETUP MUST NOT ATTEMPT TO BREAK FOR A LENGTH OF TIME
277                          EQUAL TO $USWR CONTENTS, IN SECONDS.  THE NOMINAL 16 SECONDS PER PASS, AND THE
278                          ACTUAL PASS TIME OF ABOUT 10 SECONDS (WHICH CHANGES AS TESTS MAY BE ADDED)
279                          ALLOW A SAFTY MARGIN ADDING ADDITIONAL TESTS. THE ACTUAL NUMBER OF PASSES DONE
280                          WILL BE EQUAL TO $USWR DIVIDED BY 16, (IF NOT 0, OR < 16 ).
281
282                                        !!!!!!!!!!!!!
283                          ***** THE GUARRANTEE THAT APT WILL NOT ATTEMPT TO BREAK INTO THE DIAGNOSTIC
284                          FOR A GIVEN NUMBER OF SECONDS IS PROVIDED BY THE APT SYSTEM MANAGER LOADING
285                          THE VARIABLE $PASTM WITH THE SAME NUMBER $USWR RECEIVED.
286                          IF THIS IS NOT DONE, THE MESSAGE "HUNG DIAGNOSTIC" WILL LIKELY BE RECEIVED*******
287                                        !!!!!!!!!!!!!
288
289                                 9.3  ON-BOARD ROM AND EEROM CODE
290
291                                 THESE TESTS VERIFY THE CHECKSUMS OF THE 16-BIT ROM AND THE BASE
292                                 AREA OF THE 8-BIT EEROM, AND A WRITE AND READ OF 1'S AND 0'S TO
293                                 EACH LOCATION OF THE EEROM.  THE EEROM MAY BE AFFECTED BY MULTIPLE
294                                 WRITES, (>10,000 CYCLES) SO IT IS ADVISABLE TO SELECT THIS TEST
295                                 WITH CARE.
296
297                                 THE TESTING OF THE ON-BOARD EEROM IS ACCOMPLISHED BY THE SETTING
298                                 OF BIT 12 IN THE SOFTWARE SWITCH REGISTER, WHICH WILL SEE THAT THE
299                                 EEROM TEST IS RUN ONCE.  (PASS 1)  IT SHOULD BE NOTED THAT THIS BIT
300                                 MAY HAVE ANOTHER USE IN THE SYSMAC MACRO $EOP, BUT IS NOT SO USED
301                                 WITHIN THIS PROGRAM.  THE BCSR IS ALSO SET FOR HALT ON BREAK ON
302                                 DURING THIS TEST, AS A TROUBLESHOOTING AID IN STAND-ALONE MODES.
303
```

```
                    *** WHEN RUNNING THIS TEST UNDER APT ****

                AFTER LOADING AND STARTING THE PROGRAM WITH EEROM TEST SWITCH
                ON, APT MUST NOT (BREAK) INTERRUPT THE TEST UNTIL THE TIME
                GIVEN IN SECTION 6, EXECUTION TIMES, HAS PASSSED.

                THIS TEST SHOULD NEVER BE "SCRIPTED" IN A WAY WHICH WOULD CAUSE
                THE TEST TO BE RUN MULTIPLE TIMES.  IT SHOULD BE RUN INSTEAD
                ONCE AT THE BEGINING OR END OF A SCRIPT.

                                ******************

!!      THE CONTENTS OF THE EEROM ARE LOST, EXCEPT FOR THE 109 (DECIMAL) BYTE
!!      REPRESENTING THE SETUP AREA, WHICH ARE RESTORED AT THE END OF THE EEROM
!!      SUBTEST.  IF THE TEST IS INTERRUPTED BEFORE THESE LOCATIONS ARE RE-
!!      STORED, UNPREDICTABLE RESULTS MAY OCCUR.



        9.4  LINE TIME CLOCKS CODE
        THIS PART OF THE PROGRAM VERIFIES THE FUNCTIONS OF ALL 4 CLOCK
        LINES: 3 OF THEM FROM THE SERIAL LINE CHIP (50HZ, 60HZ, 800HZ)
        AND BEVENT LINE.
        NOTE: IN UFD MODE ONLY FUNCTIONS CORRESPONDING TO BOOT ROM
        SELECTION ARE CHECKED.


        9.5  SERIAL LINE UNIT CODE
        THESE TESTS VERIFY THE FUNCTIONALITY OF THE SLU CHIP UTILIZING THE
        MAINTENANCE MODE OF THE CHIP.


        9.6  Q22BE CODE
        THESE TESTS VERIFY INTERRUPT ARBITRATION OF THE KDJ11-B, DMA
        PROTOCOL, AND CACHE FUNCTIONS RELATED TO THE DMA ACTIVITIES,
        INCLUDING PMG COUNTER.


        9.7  LIST OF SUBTESTS PERFORMED
THE FOLLOWING LIST REPRESENTS THE SEQUENTIAL ORDER OF SUBTESTS
PERFORMED IN COKDACAO..... SUBTESTS WHICH ARE SUBJECT TO THE APT
QUALIFICATIONS OF SECTION 9.2 ARE INDICATED BY A CACHE-APT LABEL.

TEST NO.        TEST NAME/FUNCTION              APT-CACHE SEL           NOTES.

TEST 1          BASE INSTRUCTION SET TESTS      .............
TEST 2          CHECK (CACHE) REGISTER ACCESS    YES
TEST 3          CCR REGISTER BIT TEST            YES
TEST 4          FORCE MISS TEST                  YES
TEST 5          HIT/MISS REGISTER TEST PART 1    YES
TEST 6          HIT/MISS REGISTER TEST           YES
TEST 7          BYTE ALLOCATION TEST             YES

TEST 10         PDR BIT 15 (BYPASS) TEST         YES
TEST 11         FLUSH CACHE TEST                 YES
TEST 12         UNCONDITIONAL BYPASS TEST        YES
TEST 13         WRITE WRONG DATA PARITY TEST     YES
```

```
TEST 14        WRITE WRONG TAG PARITY                YES
TEST 15        PARITY ABORT TEST                     YES
TEST 16        PARITY INTERRUPT TEST                 YES
TEST 17        MISCELLANEOUS PARITY TEST             YES

TEST 20      MEMORY SYSTEM ERROR REGISTER TEST       YES
TEST 21     CHECK PARITY ABORTS BLOCKED ......       YES
TEST 22        MULTI-PROCESSING INSTRUCTION          YES
TEST 23        DATA STORE RAM, TESTS                 YES
TEST 24        TAG STORE RAM TESTS                   YES
TEST 25        STANDALONE MODE TESTS                 YES
TEST 26        MOVING INVERSIONS TEST DATA RAM       YES
TEST 27        MOVING INVERSIONS FOR TAG STORE       YES

TEST 30        PCR READ/WRITE BITS                   .......
TEST 31        BCSR READ/WRITE BITS                  .......
TEST 32        16 BIT ROM CHECKSUM                   .......        HOB ON
TEST 33        8 BIT EEROM CHECKSUM TEST             .......        HOB ON
TEST 34        EEROM CHECKERBOARD MEM TEST           .......        NOTE 1.0
TEST 35        LKS BIT 7                             .......
TEST 36        LKS INTERRUPT PRIORITY                .......
TEST 37        LINE CLOCK DISABLE                    .......

TEST 40        UNCONDITIONAL CLOCK INTERUPTS         .......
TEST 41        RESETTING LKS                         .......    ONLY DONE PASS 1
TEST 42        LINE CLOCK INTERRUPTS                 YES        NOT CACHE DEP, APT BREAK SENSITIV
E
TEST 43        MAINTENANCE REGISTER TEST             .......
TEST 44        SERIAL LINE UNIT REGISTERS            .......
TEST 45        XCSR BIT 7                            .......
TEST 46        RCSR BIT 7 AND XCSR BIT 7             .......       DONE ONCE IN APT
TEST 47        RESET AND XCSR<2!0>                   .......       FIRST PASS ONLY

TEST 50        RESET AND INTERRUPT ENABLE            .......
TEST 51        INTERRUPT PRIORITY FOR SLU            .......       DONE ONCE IN APT
TEST 52        BREAK CONDITION                       .......       DONE ONLY ONCE
TEST 53        OVERRUN CONDITION                     .......       DONE ONCE IN APT
TEST 54        LED'S ON TEST                         .......         HOB ON
TEST 55        MEMORY MAPPING    NOT DONE CHAIN MODE, APT, OTHER THAN  PASS 1
TEST 56        WRONG PARITY ABORT TEST               .......
TEST 57        DMA TAG PARITY IN STANDALONE          YES             HOB DISABLED

TEST 60        DMA TAG PARITY W/O STANDALONE         YES
TEST 61        DMA WRITE HIT CYCLES                  YES
TEST 62        DIFFERENT LEVELS OF INTERRUPTS                NOT UFD, ONLY IF Q22BE FOUND
TEST 63        ARBITRATION BET PIRQ INTERRUPT                NOT UFD, ONLY IF Q22BE FOUND
TEST 64        POWER DOWN TEST                               NOT UFD, ONLY IF Q22BE FOUND
TEST 65        ARBITRATION BETWEEN INTERRUPT LEVELS          NOT UFD, ONLY IF Q22BE FOUND
TEST 66        PMG COUNTER                           YES     NOT UFD, ONLY IF TWO Q22BE FOUND

AS OF JULY 1984, THIS IS THE LIST OF SUBTESTS, ANY ADDED SUBTESTS WILL REQUIRE
EDITING OF THIS LIST....


NOTE 1.  THIS TEST WILL ONLY BE RUN IN FIRST PASS, AND THEN ONLY IF BIT 12 IS SET
         IN THE SWR.... RUNNING THIS TEST WILL DESTROY EVERYTHING IN THE EEROM
         EXCEPT THE FIRST 109 BYTES...... UFD AREA, SECONDARY BOOTS, AND LOCAL
         LANGUAGE AREA'S WOULD ALL NEED TO BE RESTORED AFTER RUNNING THIS TEST.
```

THE FIRST PASS RUN TIME MUST BE ADJUSTED ACCORDINGLY WHEN RUNNING UNDER

ε

K1

SEQ 0010

```
                                $SWR=167400
                                $SWRMK=300
                        .TITLE  COKDAC0 KDJ11-B CLUSTER DIAG.
                        ;*COPYRIGHT (C) JUL 84
                        ;*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.
                        ;*
                        $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
```

MEMORY MANAGEMENT DEFINITIONS

```
                        UIPAR4= 177650
                        UIPAR5= 177652
                        UIPAR6= 177654
                        UIPAR7= 177656
                        ;*USER "D" PAGE ADDRESS REGISTERS
                        UDPAR0= 177660
                        UDPAR1= 177662
                        UDPAR2= 177664
                        UDPAR3= 177666
                        UDPAR4= 177670
                        UDPAR5= 177672
                        UDPAR6= 177674
                        UDPAR7= 177676
                        ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
                        SIPDR0= 172200
                        SIPDR1= 172202
                        SIPDR2= 172204
                        SIPDR3= 172206
                        SIPDR4= 172210
                        SIPDR5= 172212
                        SIPDR6= 172214
                        SIPDR7= 172216
                        ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
                        SDPDR0= 172220
                        SDPDR1= 172222
                        SDPDR2= 172224
                        SDPDR3= 172226
                        SDPDR4= 172230
                        SDPDR5= 172232
                        SDPDR6= 172234
                        SDPDR7= 172236
                        ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
                        SIPAR0= 172240
                        SIPAR1= 172242
                        SIPAR2= 172244
                        SIPAR3= 172246
                        SIPAR4= 172250
                        SIPAR5= 172252
                        SIPAR6= 172254
                        SIPAR7= 172256
                        ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS
                        SDPAR0= 172260
                        SDPAR1= 172262
                        SDPAR2= 172264
                        SDPAR3= 172266
                        SDPAR4= 172270
                        SDPAR5= 172272
                        SDPAR6= 172274
                        SDPAR7= 172276
                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
                        KIPDR0= 172300
                        KIPDR1= 172302
                        KIPDR2= 172304
                        KIPDR3= 172306
                        KIPDR4= 172310
                        KIPDR5= 172312
                        KIPDR6= 172314
```

M1

MEMORY MANAGEMENT DEFINITIONS

```
172316                  KIPDR7= 172316
                        ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS
172320                  KDPDR0= 172320
172322                  KDPDR1= 172322
172324                  KDPDR2= 172324
172326                  KDPDR3= 172326
172330                  KDPDR4= 172330
172332                  KDPDR5= 172332
172334                  KDPDR6= 172334
172336                  KDPDR7= 172336
                        ;*KERNEL "I" PAGE ADDRESS REGISTERS
172340                  KIPAR0= 172340
172342                  KIPAR1= 172342
172344                  KIPAR2= 172344
172346                  KIPAR3= 172346
172350                  KIPAR4= 172350
172352                  KIPAR5= 172352
172354                  KIPAR6= 172354
172356                  KIPAR7= 172356
                        ;*KERNEL "D" PAGE ADDRESS REGISTERS
172360                  KDPAR0= 172360
172362                  KDPAR1= 172362
172364                  KDPAR2= 172364
172366                  KDPAR3= 172366
172370                  KDPAR4= 172370
172372                  KDPAR5= 172372
172374                  KDPAR6= 172374
172376                  KDPAR7= 172376
442                     .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
```

BASIC DEFINITIONS

```
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
001000                  SW9=    SW09
000400                  SW8=    SW08
000200                  SW7=    SW07
000100                  SW6=    SW06
000040                  SW5=    SW05
000020                  SW4=    SW04
000010                  SW3=    SW03
000004                  SW2=    SW02
000002                  SW1=    SW01
000001                  SW0=    SW00
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
001000                  BIT9=   BIT09
000400                  BIT8=   BIT08
000200                  BIT7=   BIT07
000100                  BIT6=   BIT06
000040                  BIT5=   BIT05
000020                  BIT4=   BIT04
```

```
                000010                  BIT3=   BIT03
                000004                  BIT2=   BIT02
                000002                  BIT1=   BIT01
                000001                  BIT0=   BIT00
                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014                  TBITVEC=14              ;;"T" BIT
                000014                  TRTVEC= 14              ;;TRACE TRAP
                000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024                  PWRVEC= 24              ;;POWER FAIL
                000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                000034                  TRAPVEC=34              ;;"TRAP" TRAP
                000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
443             000001                          UFDSET=         1               ;FLAG FOR UFD
444                                     .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
445             000200                          .=200
446 000200      005037  001160                  CLR     $TMP0
447 000204      000137  004024                  JMP     @#START
448             000220                          .=220
449 000220      012737  000777  001160          MOV     #777,$TMP0
450 000226      000137  004024                  JMP     @#START
451
452                                     .SBTTL  ACT11 HOOKS
                                        ;;**********************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000232                          $SVPC=.                 ;SAVE PC
                000046                          .=46
    000046      135546                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                          .=52
    000052      000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                000232                          .=$SVPC                 ;; RESTORE PC
453                                     .SBTTL  APT PARAMETER BLOCK
                                        ;;**********************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**********************************************************************
                000232                          .$X=.   ;;SAVE CURRENT LOCATION
                000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
    000024      000200                          200     ;;FOR APT START UP
                000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
    000044      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.
```

C2

SEQ 0015

APT PARAMETER BLOCK

```
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)
```

```
    454                             .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)
            000002                  .REPT   2
    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     207     377     377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
    001173     000
    001174     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
    001175     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
    001176     012     000          $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
    001212  000000                  $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
    001214  000000                  $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
```

APT MAILBOX-ETABLE

```
001216  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001220                          $ETABLE:                ;;APT ENVIRONMENT TABLE
001220      000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001221      000                 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001222  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001224  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001226  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001230      000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001231      000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001232  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001234      000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001235      000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001236  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001240      000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001241      000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001242  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001244      000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001245      000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001246  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001250  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001252  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001254  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001256  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001260  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001262  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001264  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001266  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001270  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001272  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001274  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001276  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001300  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001302  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001304  000000                  $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001306  000000                  $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001310  000000                  $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001312  000000                  $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001314  000000                  $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001316  000000                  $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001320  000000                  $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001322  000000                  $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
001324                          $ETEND:
```

F2

ERROR POINTER TABLE

```
                                .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                      $ERRTB:
455
456
457                             .SBTTL  ERROR DEFINITIONS
458                             ;ITEM 1
459 001324  124316                      EM1             ;CPU ERROR
460 001326  132135                      DH1             ;TEST #, ERROR PC
461 001330  133342                      DT1             ;$TMP1,$ERRPC
462 001332  000000                      0
463                             ;ITEM 2
464 001334  124352                      EM2             ;MMU ERROR
465 001336  132135                      DH1             ;TEST #, ERROR PC
466 001340  133342                      DT1             ;$TMP1,$ERRPC
467 001342  000000                      0
468                             ;ITEM 3
469 001344  124364                      EM3             ;FPP ERROR
470 001346  132135                      DH1             ;TEST #, ERROR PC
471 001350  133342                      DT1             ;$TMP1,$ERRPC
472 001352  000000                      0
473                             ;ITEM 4
474 001354  124376                      EM4             ;ERROR IN READ-WRITE BITS OF CCR
475 001356  132162                      DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
476 001360  133350                      DT4             ;$TMP1,$ERRPC,R1,CCR
477 001362  000000                      0
478                             ;ITEM 5
479 001364  124436                      EM5             ;FORCE MISS WRITES TO CACHE
480 001366  132247                      DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
481 001370  133362                      DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
482 001372  000000                      0
483                             ;ITEM 6
484 001374  124471                      EM6             ;FORCE MISS WRITE INVALIDATES CACHE
485 001376  132247                      DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
486 001400  133362                      DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
487 001402  000000                      0
488                             ;ITEM 7
489 001404  124534                      EM7             ;UNEXPECTED PARITY INTERRUPT
490 001406  132346                      DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
491 001410  133376                      DT7             ;$TMP1,$ERRPC,$BDADR,MSER
492 001412  000000                      0
493                             ;ITEM 10
494 001414  124570                      EM10            ;TAG PARITY ERROR
495 001416  132346                      DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
496 001420  133376                      DT7             ;$TMP1,$ERRPC,$BDADR,MSER
497 001422  000000                      0
498                             ;ITEM 11
499 001424  124611                      EM11            ;DATA PARITY ERROR
500 001426  132346                      DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
```

ERROR DEFINITIONS

```
001430  133376                  DT7             ;$TMP1,$ERRPC,$BDADR,MSER
001432  000000                  0
                        ;ITEM 12
001434  124633                  EM12            ;LOW BYTE PARITY ERROR
001436  132346                  DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
001440  133376                  DT7             ;$TMP1,$ERRPC,$BDADR,MSER
001442  000000                  0
                        ;ITEM 13
001444  124661                  EM13            ;HIGH BYTE PARITY ERROR
001446  132346                  DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
001450  133376                  DT7             ;$TMP1,$ERRPC,$BDADR,MSER
001452  000000                  0
                        ;ITEM 14
001454  124710                  EM14            ;ERROR DATA PATH
001456  132162                  DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
001460  133410                  DT14            ;$TMP1,$ERRPC,$GDDAT,TSTLOC
001462  000000                  0
                        ;ITEM 15
001464  124733                  EM15            ;FORCE MISS READS FROM CACHE
001466  132247                  DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
001470  133362                  DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
001472  000000                  0
                        ;ITEM 16
001474  124767                  EM16            ;FORCE MISS READS FROM CACHE AND MISS
001476  132247                  DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
001500  133362                  DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
001502  000000                  0
                        ;ITEM 17
001504  125034                  EM17            ;ERROR IN RECORDING HITS IN HIT/MISS
001506  132162                  DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
001510  133422                  DT17            ;$TMP1,$ERRPC,$GDDAT,RECDAT
001512  000000                  0
                        ;ITEM 20
001514  125100                  EM20            ;WRITE BYTE ALLOCATES CACHE
001516  132135                  DH1             ;TEST #, PC
001520  133342                  DT1             ;$TMP1,$ERRPC
001522  000000                  0
                        ;ITEM 21
001524  125133                  EM21            ;WRITE BYTE HIT DOES NOT RECORD HIT
001526  132135                  DH1             ;TEST #, PC
001530  133342                  DT1             ;$TMP1,$ERRPC
001532  000000                  0
                        ;ITEM 22
001534  125176                  EM22            ;BYTES REVERSED ON WRITE CYCLES
001536  132135                  DH1             ;TEST #, PC
001540  133342                  DT1             ;$TMP1,$ERRPC
001542  000000                  0
                        ;ITEM 23
001544  125235                  EM23            ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
001546  132247                  DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
001550  133362                  DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
001552  000000                  0
                        ;ITEM 24
001554  125311                  EM24            ;HITS RECORDED AFTER FLUSHING CACHE
001556  132421                  DH24            ;TEST #, PC, NUMBER OF HITS
001560  133434                  DT24            ;$TMP1,$ERRPC,R3
001562  000000                  0
```

H2

ERROR DEFINITIONS

```
                                ;ITEM 25
                                        EM25            ;BYPASS DOESN'T INVALIDATE CACHE
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 26
                                        EM26            ;MSER DOES NOT CLEAR ON WRITE REFERENCE
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 27
                                        EM27            ;PARITY ERROR DON'T CAUSE A MISS
                                        DH27            ;TEST #, PC, MSER, HIT/MISS
                                        DT27            ;$TMP1,$ERRPC,MSER,R3,0
                                        0
                                ;ITEM 30
                                        EM30            ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
                                        DH27            ;TEST #, PC, MSER, HIT/MISS
                                        DT27            ;$TMP1,$ERRPC,MSER,R3,0
                                        0
                                ;ITEM 31
                                        EM31            ;PARITY ERROR IGNORED
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 32
                                        EM32            ;PARITY ERROR IGNORED ON LOW BYTE
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 33
                                        EM33            ;PARITY ERROR IGNORED ON HIGH BYTE
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 34
                                        EM34            ;PARITY ABORT LOGIC DOESN'T WORK
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 35
                                        EM35            ;MSER NOT SET PROPERLY
                                        DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
                                        DT35            ;$TMP1,$ERRPC,$GDDAT,MSER,0
                                        0
                                ;ITEM 36
                                        EM36            ;PARITY INTERRUPT DOESN'T WORK
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 37
                                        EM37            ;NXM AND PARITY ABORT DIN'T HAPPEN
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 40
                                        EM40            ;PARITY ABORT NOT BLOCKED BY NXM TRAP
```

ERROR DEFINITIONS

```
    615 001716  132135                      DH1             ;TEST #, PC
    616 001720  133342                      DT1             ;$TMP1,$ERRPC
    617 001722  000000                      0
    618                             ;ITEM 41
    619 001724  126166                      EM41            ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS
    620 001726  132530                      DH41            ;TEST #, PC, INSTRUCTION OPCODE
    621 001730  133470                      DT41            ;$TMP1,$ERRPC,$BDDAT
    622 001732  000000                      0
    623                             ;ITEM 42
    624 001734  126252                      EM42            ;ERROR IN PARITY LOGIC
    625 001736  132135                      DH1             ;TEST #, PC
    626 001740  133342                      DT1             ;$TMP1,$ERRPC
    627 001742  000000                      0
    628                             ;ITEM 43
    629 001744  126300                      EM43            ;ERROR IN CACHE DATA RAMS
    630 001746  132601                      DH43            ;TEST #, PC, EXPECTED DATA, RECEIVED DATA, CACHE LOCATION
    631 001750  133500                      DT43            ;$TMP1,$ERRPC,R1,RECDAT,$BDADR
    632 001752  000000                      0
    633                             ;ITEM 44
    634 001754  126331                      EM44            ;ERROR IN NXM IN STANDALONE MODE
    635 001756  132135                      DH1             ;TEST #, PC
    636 001760  133342                      DT1             ;$TMP1,$ERRPC
    637 001762  000000                      0
    638                             ;ITEM 45
    639 001764  126371                      EM45            ;HITS NOT RECORDED PROPERLY THRU HIT/MISS
    640 001766  132135                      DH1             ;TEST #, PC
    641 001770  133342                      DT1             ;$TMP1,$ERRPC
    642 001772  000000                      0
    643                             ;ITEM 46
    644 001774  126453                      EM46            ;HIT RECORDED FOR A LOCATION NOT IN CACHE
    645 001776  132701                      DH47            ;TEST #, PC, LOCATION ACCESSED
    646 002000  133514                      DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
    647 002002  000000                      0
    648                             ;ITEM 47
    649 002004  126543                      EM47            ;MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE
    650 002006  132701                      DH47            ;TEST #, PC, LOCATION ACCESSED
    651 002010  133514                      DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
    652 002012  000000                      0
    653                             ;ITEM 50
    654 002014  126630                      EM50            ;ERROR IN TAG STORE
    655 002016  132701                      DH47            ;TEST #, PC, ADDRESS
    656 002020  133526                      DT50            ;$TMP1,$ERRPC,R1,$BDADR
    657 002022  000000                      0
    658                             ;ITEM 51
    659 002024  126653                      EM51            ;ERROR PCR READ-WRITE BITS
    660 002026  132162                      DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    661 002030  133540                      DT51            ;$TMP1,$ERRPC,$GDDAT,PCR
    662 002032  000000                      0
    663                             ;ITEM 52
    664 002034  126705                      EM52            ;ERROR IN BCSR READ-WRITE BITS
    665 002036  132162                      DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    666 002040  133552                      DT52            ;$TMP1,$ERROC,$GDDAT,BCSR
    667 002042  000000                      0
    668                             ;ITEM 53
    669 002044  126743                      EM53            ;RESET DOESN'T CLEAR BCSR<4>
    670 002046  132135                      DH1             ;TEST #, PC
    671 002050  133342                      DT1             ;$TMP1,$ERRPC
```

J2

ERROR DEFINITIONS

```
672 002052  000000                      0
673                             ;ITEM 54
674 002054  126777                      EM54            ;CHECKSUM ERROR IN 16-BIT ROM
675 002056  132135                      DH1             ;TEST #, PC
676 002060  133342                      DT1             ;$TMP1,$ERRPC
677 002062  000000                      0
678                             ;ITEM 55
679 002064  127035                      EM55            ;CHCKSUM ERROR IN 8-BIT ROM
680 002066  132135                      DH1             ;TEST #, PC
681 002070  133342                      DT1             ;$TMP1,$ERRPC
682 002072  000000                      0
683                             ;ITEM 56
684 002074  127071                      EM56            ;TIMEOUT READING LKS
685 002076  132135                      DH1             ;TEST #, PC
686 002100  133342                      DT1             ;$TMP1,$ERRPC
687 002102  000000                      0
688                             ;ITEM 57
689 002104  127115                      EM57            ;LKS<07> DOES NOT BECOME 1
690 002106  132135                      DH1             ;TEST #, PC
691 002110  133342                      DT1             ;$TMP1,$ERRPC
692 002112  000000                      0
693                             ;ITEM 60
694 002114  127147                      EM60            ;WRITE REFERENCE DOESN'T CLEAR LKS<07>
695 002116  132135                      DH1             ;TEST #, PC
696 002120  133342                      DT1             ;$TMP1,$ERRPC
697 002122  000000                      0
698                             ;ITEM 61
699 002124  127215                      EM61            ;ILLEGAL LKS INTERRUPTS
700 002126  132135                      DH1             ;TEST #, PC
701 002130  133342                      DT1             ;$TMP1,$ERRPC
702 002132  000000                      0
703                             ;ITEM 62
704 002134  127244                      EM62            ;PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>
705 002136  132135                      DH1             ;TEST #, PC
706 002140  133342                      DT1             ;$TMP1,$ERRPC
707 002142  000000                      0
708                             ;ITEM 63
709 002144  127315                      EM63            ;LKS READY DOESN'T GO LOW
710 002146  132135                      DH1             ;TEST #, PC
711 002150  133342                      DT1             ;$TMP1,$ERRPC
712 002152  000000                      0
713                             ;ITEM 64
714 002154  127346                      EM64            ;WRONG NUMBER OF LKS INTERRUPTS
715 002156  132135                      DH1             ;TEST #, PC
716 002160  133342                      DT1             ;$TMP1,$ERRPC
717 002162  000000                      0
718                             ;ITEM 65
719 002164  127405                      EM65            ;LKS INTERRUPTS HAPPEN AT WRONG PRIORITY
720 002166  132766                      DH65            ;TEST #, PC, PRIORITY
721 002170  133576                      DT65            ;$TMP1,$ERRPC,$GDDAT
722 002172  000000                      0
723                             ;ITEM 66
724 002174  127455                      EM66            ;BCSR<12> DOES NOT DISABLES LKS
725 002176  132135                      DH1             ;TEST #, PC
726 002200  133342                      DT1             ;$TMP1,$ERRPC
727 002202  000000                      0
728                             ;ITEM 67
```

K2

ERROR DEFINITIONS

```
    729 002204  127514                  EM67            ;BCSR<13> DOESN'T SET LKS<06>
    730 002206  132135                  DH1             ;TEST #, PC
    731 002210  133342                  DT1             ;$TMP1,$ERRPC
    732 002212  000000                  0
    733                         ;ITEM 70
    734 002214  127551                  EM70            ;RESET DOESN'T SET LKS<7>
    735 002216  132135                  DH1             ;TEST #, PC
    736 002220  133342                  DT1             ;$TMP1,$ERRPC
    737 002222  000000                  0
    738                         ;ITEM 71
    739 002224  127602                  EM71            ;RESET DOESN'T CLEAR LKS<06>
    740 002226  132135                  DH1             ;TEST #, PC
    741 002230  133342                  DT1             ;$TMP1,$ERRPC
    742 002232  000000                  0
    743                         ;ITEM 72
    744 002234  127636                  EM72            ;TIMEOUT READING SLU REGISTERS
    745 002236  133032                  DH72            ;TEST #, PC, ADDRESS FAILED
    746 002240  133470                  DT41            ;$TMP1,$ERRPC,$BDDAT
    747 002242  000000                  0
    748                         ;ITEM 73
    749 002244  127674                  EM73            ;XMIT READY DIDN'T GO LOW
    750 002246  132135                  DH1             ;TEST #, PC
    751 002250  133342                  DT1             ;$TMP1,$ERRPC
    752 002252  000000                  0
    753                         ;ITEM 74
    754 002254  127720                  EM74            ;RCSR DOESN'T BECOME 1
    755 002256  132135                  DH1             ;TEST #, PC
    756 002260  133342                  DT1             ;$TMP1,$ERRPC
    757 002262  000000                  0
    758                         ;ITEM 75
    759 002264  127751                  EM75            ;WRONG CHARACTER RECEIVED
    760 002266  132162                  DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    761 002270  133606                  DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
    762 002272  000000                  0
    763                         ;ITEM 76
    764 002274  130002                  EM76            ;RCSR<07> NOT CLEARED AFTER READING RBUF
    765 002276  132135                  DH1             ;TEST #, PC
    766 002300  133342                  DT1             ;$TMP1,$ERRPC
    767 002302  000000                  0
    768                         ;ITEM 77
    769 002304  130052                  EM77            ;XCSR<07> NOT SET ON RESET
    770 002306  132135                  DH1             ;TEST #, PC
    771 002310  133342                  DT1             ;$TMP1,$ERRPC
    772 002312  000000                  0
    773                         ;ITEM 100
    774 002314  130104                  EM100           ;RCSR<07> NOT CLEARED ON RESET
    775 002316  132135                  DH1             ;TEST #, PC
    776 002320  133342                  DT1             ;$TMP1,$ERRPC
    777 002322  000000                  0
    778                         ;ITEM 101
    779 002324  130142                  EM101           ;SLU INTERRUPTS HAPPEN AT 4
    780 002326  132135                  DH1             ;TEST #, PC
    781 002330  133342                  DT1             ;$TMP1,$ERRPC
    782 002332  000000                  0
    783                         ;ITEM 102
    784 002334  130175                  EM102           ;RESET DOES NOT CLEAR XCSR<6> AND RSCR<6>
    785 002336  132135                  DH1             ;TEST #, PC
```

L2

ERROR DEFINITIONS

```
786 002340  133342                      DT1             ;$TMP1,$ERRPC
787 002342  000000                      0
788                             ;ITEM 103
789 002344  130257                      EM103           ;TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>
790 002346  132135                      DH1             ;TEST #, PC
791 002350  133342                      DT1             ;$TMP1,$ERRPC
792 002352  000000                      0
793                             ;ITEM 104
794 002354  130332                      EM104           ;RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>
795 002356  132135                      DH1             ;TEST #, PC
796 002360  133342                      DT1             ;$TMP1,$ERRPC
797 002362  000000                      0
798                             ;ITEM 105
799 002364  130402                      EM105           ;BREAK CONDITION DOES NOT SET RBUF PROPERLY
800 002366  133076                      DH105           ;TEST #, PC, RBUF
801 002370  133620                      DT105           ;$TMP1,$ERRPC,RBUF
802 002372  000000                      0
803                             ;ITEM 106
804 002374  130455                      EM106           ;RBUF WASN'T CLEARED ON NEXT CHAR.
805 002376  133076                      DH105           ;TEST #, PC, RBUF
806 002400  133620                      DT105           ;$TMP1,$ERRPC,RBUF
807 002402  000000                      0
808                             ;ITEM 107
809 002404  130533                      EM107           ;ERROR IN WRITING TO XCSR<0>
810 002406  132135                      DH1             ;TEST #, PC
811 002410  133342                      DT1             ;$TMP1,$ERRPC
812 002412  000000                      0
813                             ;ITEM 110
814 002414  130567                      EM110           ;RESET DOES NOT CLEAR XCSR<00>
815 002416  132135                      DH1             ;TEST #, PC
816 002420  133342                      DT1             ;$TMP1,$ERRPC
817 002422  000000                      0
818                             ;ITEM 111
819 002424  130631                      EM111           ;FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND
820 002426  132162                      DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
821 002430  133606                      DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
822 002432  000000                      0
823                             ;ITEM 112
824 002434  130707                      EM112           ;OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF
825 002436  133076                      DH105           ;TEST #, PC, RBUF
826 002440  133620                      DT105           ;$TMP1,$ERRPC,RBUF
827 002442  000000                      0
828                             ;ITEM 113
829 002444  130772                      EM113           ;RBUF WAS NOT CLEARED ON THE NEXT CHARACTER
830 002446  133076                      DH105           ;TEST #, PC, RBUF
831 002450  133620                      DT105           ;$TMP1,$ERRPC,RBUF
832 002452  000000                      0
833                             ;ITEM 114
834 002454  131056                      EM114           ;ERROR IN XCSR<2>
835 002456  132135                      DH1             ;TEST #, PC
836 002460  133342                      DT1             ;$TMP1,$ERRPC
837 002462  000000                      0
838                             ;ITEM 115
839 002464  131077                      EM115           ;ERROR IN TAG STORE FROM STANDALONE MODE
840 002466  133130                      DH115           ;TEST #, PC, MSER, ADDRESS ACCESSED
841 002470  133630                      DT115           ;$TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR
842 002472  000000                      0
```

M2

ERROR DEFINITIONS

```
843
844 002474  131147                ;ITEM 116
                                        EM116           ;DMA TAG PARITY DOES NOT SET MSER<4>
845 002476  132135                      DH1             ;TEST #, PC
846 002500  133342                      DT1             ;$TMP1,$ERRPC
847 002502  000000                      0
848                                 ;ITEM 117
849 002504  131230                      EM117           ;MSER<13>NOT SET IN STANDALONE MODE
850 002506  132135                      DH1             ;TEST #, PC
851 002510  133342                      DT1             ;$TMP1,$ERRPC
852 002512  000000                      0
853                                 ;ITEM 120
854 002514  131273                      EM120           ;DMA WRITE HITS DON'T INVALIDATE CACHE
855 002516  132135                      DH1             ;TEST #, PC
856 002520  133342                      DT1             ;$TMP1,$ERRPC
857 002522  000000                      0
858                                 ;ITEM 121
859 002524  131341                      EM121           ;IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED
860 002526  132135                      DH1             ;TEST #, PC
861 002530  133342                      DT1             ;$TMP1,$ERRPC
862 002532  000000                      0
863                                 ;ITEM 122
864 002534  131437                      EM122           ;READ DMA HIT IS MESSED UP
865 002536  132135                      DH1             ;TEST #, PC
866 002540  133342                      DT1             ;$TMP1,$ERRPC
867 002542  000000                      0
868                                 ;ITEM 123
869 002544  131465                      EM123           ;ERROR IN DMA CYCLES FROM Q22BE
870 002546  132135                      DH1             ;TEST #, PC
871 002550  133342                      DT1             ;$TMP1,$ERRPC
872 002552  000000                      0
873                                 ;ITEM 124
874 002554  131517                      EM124           ;PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS
875 002556  132135                      DH1             ;TEST #, PC
876 002560  133342                      DT1             ;$TMP1,$ERRPC
877 002562  000000                      0
878                                 ;ITEM 125
879 002564  131611                      EM125           ;NO POWER DOWN TRAP TO 24 OCCUR
880 002566  132135                      DH1             ;TEST #, PC
881 002570  133342                      DT1             ;$TMP1,$ERRPC
882 002572  000000                      0
883                                 ;ITEM 126
884 002574  131650                      EM126           ;ERROR IN INTERRUPTS FROM Q22BE
885 002576  132135                      DH1             ;TEST #, PC
886 002600  133342                      DT1             ;$TMP1,$ERRPC
887 002602  000000                      0
888                                 ;ITEM 127
889 002604  131705                      EM127           ;ERROR IN PMG COUNTER
890 002606  132135                      DH1             ;TEST #, PC
891 002610  133342                      DT1             ;$TMP1,$ERRPC
892 002612  000000                      0
893                                 ;ITEM 130
894 002614  131747                      EM130           ;UNEXPECTED TIMEOUT
895 002616  132135                      DH1             ;TEST #, PC
896 002620  133644                      DT130           ;$TMP1,$ERRPC
897 002622  000000                      0
898                                 ;ITEM 131
899 002624  131774                      EM131           ;ERROR WRITING TO LKS<6>
```

N2

ERROR DEFINITIONS

```
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 132
                                        EM132           ;MAINTENANCE REGISTER ERROR
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 133
                                        EM133           ; EEROM TYPE ERROR
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 134
                                        EM134           ; EROM WRITE/READ ERROR
                                        DH134           ;TEST #, PC, ERROR COUNT, PATTERN, DATA READ, ADDRESS
                                        DT134           ;$TMP1,$ERRPC,$TMP0,R3,$BDDAT,$BDADR
                                        0

                                .SBTTL  GLOBAL VARIABLES AND REGISTER NAMES

                                ;REGISTERS FOR THE FIRST Q22BE
                                CSR1:   .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
                                CSR2:   .WORD   0               ;CONTROL/STATUS REGISTER 2
                                BA:     .WORD   0               ;DMA ADDRESS FOR Q22BE
                                WC:     .WORD   0               ;WORD COUNT REGISTER
                                DATA:   .WORD   0               ;DMA DATA FOR Q22BE
                                VQBE1:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
                                VQPR1:  .WORD   0               ;PRIORITY
                                SIMGOA: .WORD   0               ;SIMULTANEUOS GO ADDRESS REGISTER

                                ;REGISTERS FOR THE SECOND Q22BE
                                CSR12:  .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
                                CSR22:  .WORD   0               ;CONTROL/STATUS REGISTER 2
                                BA2:    .WORD   0               ;DMA ADDRESS FOR Q22BE
                                WC2:    .WORD   0               ;WORD COUNT REGISTER
                                DATA2:  .WORD   0               ;DMA DATA FOR Q22BE
                                VQBE2:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
                                VQPR2:  .WORD   0               ;PRIORITY

                                LKSFL:  .WORD   0
                                ACTCHS: .WORD   0               ;ACTUAL CHECKSUM
                                SAVPCR: .WORD   0
                                SAVBR:  .WORD   0
                                .=2740
                                TEMP:   .WORD   0
                                        .BLKW   15.             ;RESERVED FOR BLOCK MODE TRANSFER
                                TIMOUT: .WORD   0
                                Q22EN:  .WORD   32,12,6,2       ;PRIORITY 7-4 FOR Q22BE


                        BCR=            177524  ;BOOT/DIAGNOSTICS CONFIGURATION
                        BDR=            177524  ;BOOT/DIAGNOSTICS DISPLAY
                        BCSR=           177520  ;BOOT/DIAGNOSTICS STATUS
                        CCR=            177746  ;CACHE CONTROL REGISTER
                        HITMIS=         177752  ;HIT OR MISS REGISTER
```

```
        177734                  KMCR=           177734          ;UNIBUS CONFIGURATION REGISTER
        177546                  LKS=            177546          ;CLOCK STATUS REGISTER
        177750                  MAIREG=         177750          ;MAINTENANCE REGISTER
        177744                  MSER=           177744          ;MEMORY SYSTEM ERROR
        177522                  PCR=            177522          ;PAGE CONTROL REGISTER
        177772                  PIR=            177772          ;PROGRAM INTERRUPT REQUEST
        177562                  RBUF=           177562          ;RECEIVER DATA BUFFER
        177560                  RCSR=           177560          ;RECEIVER STATUS REGISTER
        177566                  XBUF=           177566          ;TRANSMITTER DATA BUFFER
        177564                  XCSR=           177564          ;TRANSMITTER STATUS REGISTER

        177766                  CPEREG=         177766          ;CPU ERROR REGISTER

        177572                  MMR0=SR0                        ;MEMORY MANAGEMENT REG. 0
        177574                  MMR1=SR1                        ;MEMORY MANAGEMENT REG. 1
        177576                  MMR2=SR2                        ;MEMORY MANAGEMENT REG. 2
        172516                  MMR3=SR3                        ;MEMORY MANAGEMENT REG. 3
        120001                  POLY= 120001
        000000                  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.
003012  000000          SLOC00: .WORD   0
003014  000000          SLOC01: .WORD   0

                        ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE WORKING DATA.
003016  000000          LOWADD: .WORD   0               ;STORES LOW ADDRESS FOR RAM TESTS
003020  000000          GOODAD: .WORD   0               ;STORES GOOD ADDRESS FOR RAM TESTS
003022  000000          ERRCNT: .WORD   0               ; CONTAINS TOTAL NO. OF EEROM ERRORS
003024  000000          TSTADD: .WORD   0               ;ADDRESS STORE FOR RAM TESTS
003026  000000          NEWADD: .WORD   0               ;ADDRESS STORE FOR RAM TEST
003030  000000          FLAG:   .WORD   0               ;USED TO STORE "FLAG" CONDITIONS
003032  000000          CCHPAS: .WORD   0               ; FLAG-COUNTER FOR CONTROL OF CACHE SUBTESTS
003034  000000          EEPAS:  .WORD   0               ; FLAG-COUNTER FOR CONTROL OF EEROM SUBTEST
003036  000000          SAVSUP: .WORD   0               ;USED TO STORE SUPERVISOR STACK VALUE
003040  000000          SAVUSE: .WORD   0               ;USED TO STORE USER STACK VALUE
003042  000000          SAVMR0: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 0 DATA
003044  000000          SAVMR1: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 1 DATA
003046  000000          SAVMR2: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 2 DATA
003050  000000          SAVSWR: .WORD   0               ; SAVE SFTWRE SWTCH REG DURING EEROM TEST
003052                  FLOAT:  .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
003062                  FLO:    .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
003072  000000          SEQ:    .WORD   0               ;STORES SEQUENCE NUMBER FOR JUMP TESTS
003074  000000          SPS:    .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
003076  000000          SPSJ:   .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
003100                  BTEXP:  .BLKW   4               ;STORES EXPONENT DURING BIT TESTS
003110                  BTRES:  .BLKW   4               ;STORES RECIEVED DATA FOR BIT TESTS
003120  000000          COUNT:  .WORD   0               ;ERROR INDICATOR FOR FLOATING POINT TESTS
003122                  RECFEC: .BLKW   4               ;RECIEVED FLOATING POINT EXCEPTION CODE
```

GLOBAL DATA SECTION

```
1013 003132                            RECST:  .BLKW   4               ;RECIEVED FLOATING POINT STATUS
1014 003142                            RECDST: .BLKW   4               ;DESTINATION ADDRESS FOR FLOATING POINT TESTS
1015
1016                                   ;THESE LOCATIONS ARE USED BY MORE THAN ONE TEST AS LOOP COUNTERS
1017 003152  000000                    ALLCTR: .WORD   0
1018 003154  000000                    LOOPIN: .WORD   0
1019
1020                                   ;SOME MORE TEMPORARY STORAGE FOR RAM TESTS
1021 003156  000000                    SAVPOS: .WORD   0               ;STORES TEMPORARY BIT POSITIONS FOR RAM TESTS
1022 003160  000000                    MASK:   .WORD   0               ;STORES BIT MASK FOR ERROR ISOLATION
1023
1024                                   ;!!!!!THIS IS IT. THE PROGRAM TEST LOCATION!!!!!!!!!!!!!!!!!!
1025 003162  000000                    TSTLOC: .WORD   0
1026 003164                                    .BLKW   20.
1027                                           ;FPP REGISTER DEFINITIONS
1028         000000                            AC0= %0
1029         000001                            AC1= %1
1030         000002                            AC2= %2
1031         000003                            AC3= %3
1032         000004                            AC4= %4
1033         000005                            AC5= %5
1034         000006                            AC6= %6
1035         000007                            AC7= %7
1036
1037                                           ;FPP INTERRUPT VECTOR
1038
1039         000244                            FPVEC=244
1040
1041
1042         001000                                    STBOT=  1000
1043
1044
1046 003234  123456                    TAB1:   .WORD   123456
1047 003236  000000                            .WORD   000000
1048 003240  000000                            .WORD   0
1049 003242  000001                            .WORD   1
1050 003244  055555                    TAB2:   .WORD   055555
1051 003246  177777                            .WORD   177777
1052 003250  145671                            .WORD   145671
1053 003252  100000                            .WORD   100000
1054 003254  003000                    TAB3:   .WORD   003000
1055 003256  123456                            .WORD   123456
1056 003260  000000                            .WORD   0
1057 003262  000000                            .WORD   0
1058 003264  055555                    TAB4:   .WORD   55555
1059 003266  177777                            .WORD   -1
1060 003270  000000                            .WORD   0
1061 003272  000000                            .WORD   0
1062 003274  043243                    TAB5:   .WORD   43243
1063 003276  000000                            .WORD   0
1064 003300  000000                            .WORD   0
1065 003302  000000                            .WORD   0
1066 003304  162400                    TAB5A:  .WORD   162400
1067 003306  000000                            .WORD   0
1068 003310  000000                            .WORD   0
1069 003312  000000                            .WORD   0
1070 003314  000000                    TAB6:   .WORD   0
```

GLOBAL DATA SECTION

```
1071 003316  000000                          .WORD   0
1072 003320  000000                          .WORD   0
1073 003322  000000                          .WORD   0
1074 003324  047050                TAB6A:    .WORD   47050
1075 003326  010000                          .WORD   10000
1076 003330  000000                          .WORD   0
1077 003332  000000                          .WORD   0
1078 003334  000200                TAB7:     .WORD   200
1079 003336  000000                          .WORD   0
1080 003340  000000                          .WORD   0
1081 003342  000000                          .WORD   0
1082 003344  000200                TAB8:     .WORD   200
1083 003346  000000                          .WORD   0
1084 003350  000000                          .WORD   0
1085 003352  000001                          .WORD   1
1086 003354  000400  000000  000000  TAB9:     .WORD   400,0,0,0
     003362  000000
1087 003364  030000                TAB10:    .WORD   30000
1088 003366  003000                          .WORD   3000
1089 003370  000000                          .WORD   0
1090 003372  000000                          .WORD   0
1091 003374  016400                TAB11:    .WORD   16400
1092 003376  000000                          .WORD   0
1093 003400  000000                          .WORD   0
1094 003402  000000                          .WORD   0
1095 003404  030000  003000  000002  TAB11A:  .WORD   30000,3000,2,0
     003412  000000
1096 003414  016100  000000  000000  TAB12:    .WORD   16100,0,0,1
     003422  000001
1097 003424  016200                TAB13:    .WORD   16200
1098 003426  000000                          .WORD   0
1099 003430  000000                          .WORD   0
1100 003432  000001                          .WORD   1
1101 003434  030000  003000  000000  TAB13B:  .WORD   30000,3000,0,140000
     003442  140000
1102 003444  030000                TAB14:    .WORD   30000
1103 003446  000000                          .WORD   0
1104 003450  000000                          .WORD   0
1105 003452  000000                          .WORD   0
1106 003454  024700                TAB15:    .WORD   24700
1107 003456  000000                          .WORD   0
1108 003460  000000                          .WORD   0
1109 003462  000000                          .WORD   0
1110 003464  025000                TAB16:    .WORD   25000
1111 003466  175363                          .WORD   175363
1112 003470  123456                          .WORD   123456
1113 003472  123456                          .WORD   123456
1114 003474  030000                TAB17:    .WORD   30000
1115 003476  007020                          .WORD   7020
1116 003500  000000  000000                  .WORD   0,0
1117 003504  023456                TAB18:    .WORD   23456
1118 003506  000000                          .WORD   0
1119 003510  000000                          .WORD   0
1120 003512  000001                          .WORD   1
1121 003514  100200  000000  000000  TAB21:    .WORD   100200,0,0,0
     003522  000000
1122 003524  100400  000000  000000  TAB22:    .WORD   100400,0,0,0
```

GLOBAL DATA SECTION

```
                003532  000000
1123            003534  000200  000000  000000  TAB23:  .WORD   200,0,0,1
                003542  000001
1124            003544  062400  000000  000000  TAB24:  .WORD   62400,0,0,0
                003552  000000
1125            003554  001100  000000  000000  TAB25:  .WORD   1100,0,0,0
                003562  000000
1126            003564  100600  000000  000000  TAB26:  .WORD   100600,0,0,0
                003572  000000
1127            003574  001000  000000  000000  TAB27:  .WORD   1000,0,0,0
                003602  000000
1128            003604  000600  000000  000000  TAB28:  .WORD   600,0,0,0
                003612  000000
1129            003614  010100  000000  000000  TAB29:  .WORD   10100,0,0,0
                003622  000000
1130            003624  010100  000000  002000  TAB29A: .WORD   10100,0,2000,0
                003632  000000
1131                                            ;
1132            003634  000500  000000  000000  TAB30:  .WORD   500,0,0,0
                003642  000000
1133            003644  100400  000000  000000  TAB31:  .WORD   100400,0,0,0
                003652  000000
1134            003654  016000  000000  000000  TAB32:  .WORD   16000,0,0,0
                003662  000000
1135            003664  011600  000000  000000  TAB33:  .WORD   11600,0,0,0
                003672  000000
1136            003674  000640  000000  000000  TAB34:  .WORD   640,0,0,0
                003702  000000
1137            003704  077600  000000  000000  TAB40:  .WORD   77600,0,0,0
                003712  000000
1138            003714  100200  000000  000000  TAB41:  .WORD   100200,0,0,1
                003722  000001
1139            003724  000340  000000  000000  TAB42:  .WORD   340,0,0,0
                003732  000000
1140            003734  000077  177777  177777  TAB43:  .WORD   77,177777,177777,177776
                003742  177776
1141            003744  000577  177777  177777  TAB45:  .WORD   577,-1,-1,-1
                003752  177777
1142            003754  000577  177777  000000  TAB46:  .WORD   577,-1,0,0
                003762  000000
1143            003764  173737  124242  052525  TAB47:  .WORD   173737,124242,052525,12346
                003772  012346
1144            003774  000000  000000  052525  TAB47A: .WORD   0,0,052525,12346
                004002  012346
1145            004004  173737  124242  000000  TAB48:  .WORD   173737,124242,0,0
                004012  000000
1146            004014  000600  000000  000000  TAB49:  .WORD   600,0,0,0
                004022  000000
1147
1148            004024                          START:
                                                ;; LCP/ORION ROUTINE TO SAVE EMTULATOR AND PRIORITY
                004024  005737  004112          EMTSAV: TST     SAV30           ;; FIRST TIME THROUGH ?
                004030  001034                          BNE     VMKOR           ;; BRANCH IF BEEN HERE ALREADY
                004032  032737  000040  000052          BIT     #BIT5,@#52      ;; ARE WE IN UFD MODE ?
                004040  001430                          BEQ     VMKOR           ;; LEAVE IF NOT
                004042  012737  177777  004116          MOV     #-1,UFDFLG      ;; SET UFD FLAG
                004050  032737  000100  000052          BIT     #BIT6,@#52      ;; ARE WE IN QUIET MODE ?
```

GLOBAL DATA SECTION

```
        004056  001403                          BEQ     1$                      ;; BR IF NOT
        004060  012737  177777  004120          MOV     #-1,UQUIET              ;; SET QUIET MODE
        004066  104042                  1$:     EMT     42                      ;; GET ADDRESS OF XXDP DCA TABLE
        004070  005060  000042                  CLR     42(R0)                  ;; CLR XXDP+ "DRSERR"
        004074  013737  000030  004112          MOV     30,SAV30                ;; SAVE EMULATOR ADDRESS
        004102  013737  000032  004114          MOV     32,SAV32                ;; SAVE EMULATOR PRIORITY LEVEL
        004110  000404                          BR      VMKOR                   ;; GET AROUND TAG AREA
        004112  000000                  SAV30:  .WORD   0                       ;; PUT EMULATOR INFO HERE
        004114  000000                  SAV32:  .WORD   0                       ;; PUT PRIORITY LOCATION        HERE
        004116  000000                  UFDFLG: .WORD   0                       ;; USER FRIENDLY MODE FLAG
        004120  000000                  UQUIET: .WORD   0                       ;; UFD QUIET MODE FLAG
        004122                          VMKOR:
                                        ;*******************************************************************************
   1149 004122                          1$:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
        004122  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
        004126  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
        004130  022706  001140                  CMP     #SWR,R6 ;;DONE?
        004134  001374                          BNE     .-6             ;;LOOP BACK IF NO
        004136  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
        004142  012737  135602  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
        004150  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
        004156  012737  136104  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
        004164  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
        004172  012737  140630  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
        004200  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
        004206  012737  140712  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
        004214  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
        004222  013737  135514  135506          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
        004230  005037  001164                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
        004234  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
        004240  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
        004246  012737  004246  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
        004254  012737  004254  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
        004262  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
        004266  012737  004322  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
        004274  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
        004302  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
        004310  022777  177777  174622          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
        004316  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
        004320  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
        004322  012716  004330          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
        004326  000002                          RTI
        004330  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
        004336  012737  000174  001142          MOV     #DISPREG,DISPLAY
        004344  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
        004350  005037  001206                  CLR     $PASS           ;;CLEAR PASS COUNT
        004354  132737  000200  001221          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
        004362  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
        004364  012737  001222  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
        004372                          67$:
   1150 004372  013737  004116  004120          MOV     UFDFLG,UQUIET   ;ABORT IN UFD ON ERROR
```

```
INITIALIZE THE COMMON TAGS

 1151 004400  012737  135056  000004          MOV     #TOUT,@#ERRVEC  ;POINT TO TIMEOUT ROUTINE
 1152 004406  012737  000340  000006          MOV     #340,@#ERRVEC+2 ;AT PRIORITY 7
 1153 004414  012737  134326  000114          MOV     #RAMPAR,@#114   ;POINT PARITY ABORT
 1154 004422  012737  000340  000116          MOV     #340,@#116      ;AT PRIORITY7
 1155 004430  012737  135240  000250          MOV     #MMUTRP,@#250   ;POINT MMU TRAP VECTOR
 1156 004436  012737  000340  000252          MOV     #340,@#252      ;
 1157 004444  005037  177766                  CLR     @#177766        ;CLEAR CPU ERROR REGISTER
 1158 004450  032737  000100  000052          BIT     #BIT06,@#52     ;IN UFD QUIET MODE ?
 1159 004456  001122                          BNE     LOOP            ;IF SO, SKIP PRINTOUT
 1160 004460  012701  004472                  MOV     #.+12,R1        ;STORE LOCATION POINTER
 1161 004464  012711  177777                  MOV     #-1,(R1)        ;MAKE SURE TO PRINT NOT ONLY AT FIRST TIME
 1162                                 .SBTTL  TYPE PROGRAM NAME
                                      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
      004470  005227  177777                  INC     #-1             ;;FIRST TIME?
      004474  001044                          BNE     68$             ;;BRANCH IF NO
      004476  022737  135546  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
      004504  001440                          BEQ     68$             ;;BRANCH IF YES
      004506  104401  004554                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                                   .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
      004512  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
      004516  001012                          BNE     70$             ;;BRANCH IF YES
      004520  123727  001220  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
      004526  001406                          BEQ     70$             ;;BRANCH IF YES
      004530  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
      004536  001005                          BNE     71$             ;;BRANCH IF NO
      004540  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
      004542  000403                          BR      71$
      004544  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
      004552                       71$:
      004552  000415                          BR      68$             ;;GET OVER THE ASCIZ
                                   ;;69$:  .ASCIZ  <CRLF>#KDJ11-B CPU DIAGNOSTIC#<CRLF>
      004606                       68$:
 1163
 1164 004606  000240                          NOP                     ; DEBUG AID
 1165 004610  123727  001220  000001          CMPB    $ENV,#1         ; RUNNING UNDER APT?
 1166 004616  001020                          BNE     20$             ; DEFAULT NO-APT INITIALIZATION
 1167 004620  013700  001224                  MOV     $USWR,R0        ; WORK COPY PASS CALCULATION
 1168 004624  005700                          TST     R0              ; IF = 0 DEFAULT VALUE
 1169 004626  001420                          BEQ     25$             ; SETUP DEFAULT VALUE
 1170
 1171 004630  042700  000017                  BIC     #17,R0          ; CLEAR LOW ORDER NIBBLE
 1172 004634  000241                          CLC                     ; ASSURE NO UNKNOWNS
 1173 004636  006000                          ROR     R0              ; 4 ROTATES = DIVIDE
 1174 004640  006000                          ROR     R0              ; BY 16 (=PASS TIME)
 1175 004642  006000                          ROR     R0              ; THIS AREA SUBROUTINED
 1176 004644  006000                          ROR     R0              ; WITH GENERAL PURPOSE
 1177 004646  005700                          TST     R0              ; DIVIDE, THIS TEST TO
 1178 004650  001413                          BEQ     30$             ; DETERMINE SKIP ALTOGETHER
 1179
 1180 004652  010037  003032                  MOV     R0,CCHPAS       ;RESIDUE = NO. OF DESIRED PASSES
 1181 004656  000413                          BR      35$             ; CONTINUE ON
 1182 004660  012737  177777  003032  20$:    MOV     #-1,CCHPAS      ; LARGEST NUMBER NO APT MODE??
 1183 004666  000407                          BR      35$
 1184 004670  012737  000001  003032  25$:    MOV     #1,CCHPAS       ; NORMAL DEFAULT = 1
 1185 004676  000403                          BR      35$
 1186 004700  012737  000000  003032  30$:    MOV     #0,CCHPAS       ; NO CACHE TESTS INCLUDED
 1187 004706  000240                  35$:    NOP                     ; DEBUG AID
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
1188
1189 004710  032737  000200  000052          BIT     #BIT07,@#52     ;UFD MODE?
1190 004716  001002                          BNE     LOOP            ;IF YES, BRANCH
1191 004720  004737  134404                  JSR     PC,Q22SIZ       ;SIZE FOR Q22BE
1192 004724                          LOOP:
1193                                 .DSABLE AMA
1195                                 ;;*****************************************************************
     004724  000004                  TST1:   SCOPE
1196                                 .SBTTL  BASE INSTRUCTION SET TESTS
1197                                 ;*************************************************************************
1198                                 ;*************************************************************************
1199                                 ;                       BEGIN BASE INSTRUCTION SET TESTING
1200                                 ;*************************************************************************
1201                                 ;*************************************************************************
1210 004726                          FRSTST:
1211
1212                                 ;TEST BEQ BNE INSTRUCTIONS
1213                                 ;;*****************************************************************
                                     ;THESE TWO INSTRUCTIONS ARE FUNDAMENTAL TO RECOGNIZING ERROR CONDITIONS
1214 004726  000277                          SCC
1215 004730  000244                          CLZ                             ;CC=0100 - Z BIT CLEARED
1216 004732  001401                          BEQ     1$                      ;*TEST INSTR -TRY TO CAUSE A BEQ ERROR
1217 004734  001001                          BNE     2$                      ;BRANCH IF GOOD
1218                                                                         ;THE Z FLAG DIDNT CLEAR OR BRANCH FAILED.
1219                                                                         ;FAILURE AT THIS LOCATION
1220                                                                         ;COULD MEAN A BUS PROBLEM, MICRO-CODE PROBLEM
1221                                                                         ;CONDITION CODE PROBLEM OR JUST ABOUT ANYTHING
1222                                                                         ;ELSE.
1223 004736  104001                  1$:     ERROR   .+1                     ;CPU ERROR
1224 004740  000257                  2$:     CCC                             ;
1225 004742  000264                          SEZ                             ;COND CODES = 0100 (ZERO)
1226 004744  001001                          BNE     3$                      ;*TEST INSTR* TRY TO BRANCH ON ZERO FLAG
1227 004746  001401                          BEQ     4$                      ;*TEST INSTR* BRANCH IF GOOD
1228                                                                         ;BRANCH FAILURE WITH Z BIT SET
1229 004750  104001                  3$:     ERROR   .+1                     ;CPU ERROR
1230 004752                          4$:                                     ;END OF TEST
1240 004752                          M2:
1241
1242                                 ;       TEST BRANCH ON CARRY
1243                                 ;;*****************************************************************
                                     ;THIS IS A TEST TO SEE IF THE MODULE FORM ANTICIPATED IS FEASIBLE.
1244 004752  000257                          CCC                             ;CC=0000
1245 004754  103001                          BCC     2$                      ;*TEST INSTR*
1246                                                                         ;BRANCH CARRY CLEAR FAILED
1247 004756  104001                  1$:     ERROR   .+1                     ;CPU ERROR
1248 004760  000261                  2$:     SEC                             ;CC=1111
1249 004762  103401                          BCS     4$                      ;*TEST INSTR*
1250                                                                         ; BRANCH CARRY SET FAILED
1251 004764  104001                  3$:     ERROR   .+1                     ;CPU ERROR
1252 004766                          4$:
1253
1256 004766                          M3:
1257
1258                                 ;       TEST DATA PATHS
1259 004766  005000                          CLR     R0
1260 004770  001005                          BNE     1$                      ;TRY TO INSURE WE ARE TESTING
1261                                                                         ;THE DATA PATH AND NOT THE "CLR R0" INSTRUCTION
```

BASE INSTRUCTION SET TESTS

```
1262 004772  005010                        CLR     (R0)            ;FORCE LOCATION TO ZERO
1263 004774  001003                        BNE     1$              ;TRY TO INSURE 0=0
1264 004776  005737  000000                TST     @#0             ;AGAIN, TRY TO INSURE THAT 0=0
1265 005002  001401                        BEQ     2$              ;BRANCH IF GOOD
1266                                                               ;LOCATION 0 NOT SETUP PROPERLY
1267 005004  104001                1$:     ERROR   +1              ;CPU ERROR
1268 005006                        2$:
1271 005006                        M4:
1272
1273                               ;       TEST DATA PATHS - ONES AND ZEROS
1274 005006  012737  125252  000000        MOV     #125252,@#0             ;0=125252
1275 005014  022737  125252  000000        CMP     #125252,@#0             ;SEE IF DATA MADE IT
1276 005022  001401                        BEQ     2$                      ;BRANCH IF IF DATA IS GOOD
1277                                                                       ;ERROR! EITHER THE BUS IS BAD,
1278                                                                       ;OR THE MOV OR COMPARE
1279                                                                       ;INSTRUCTIONS FAILED
1280 005024  104001                1$:     ERROR   +1              ;CPU ERROR
1281 005026                        2$:                             ;END OF TEST
1282
1283                               ;
1286 005026                        M5:
1287
1288                               ;       TEST DATA PATHS - DATA 0'S AND 1'S
1289 005026  012737  052525  000000        MOV     #052525,@#0             ;SETUP DATA
1290 005034  023727  000000  052525        CMP     @#0,#052525             ; TEST FOR CORRECT DATA
1291 005042  001401                        BEQ     2$
1292 005044  104001                1$:     ERROR   +1              ;CPU ERROR
1293 005046                        2$:
1294                               ;
1297 005046                        M6:
1298                               ;       TEST DATA PATHS - 1'S
1299 005046  005037  000000                CLR     @#0
1300 005052  005137  000000                COM     @#0             ;SET UP MEMORY LOCATION 0 = 111111
1301 005056  023727  000000  177777        CMP     @#0,#177777     ; TEST DATA
1302 005064  001401                        BEQ     2$              ;BRANCH IF NO ERROR
1303 005066  104001                1$:     ERROR   +1              ;CPU ERROR
1304 005070                        2$:
1305
1306                               ;
1309 005070                        GPROTS:
1310                               ;       R0 BIT TESTS
1311 005070  012700  177777                MOV     #177777,R0              ;R0=177777
1312 005074  020027  177777                CMP     R0,#177777              ;DOES R0=177777
1313 005100  001401                        BEQ     1$                      ;YES GO ON
1314                                                                       ;NO GO TO ERROR
1315 005102  104001                        ERROR   +1              ;CPU ERROR
1316 005104  005000                1$:     CLR     R0                      ;R0=0
1317 005106  020027  000000                CMP     R0,#0                   ;DOES R0=0
1318 005112  001401                        BEQ     2$                      ;YES GO ON
1319                                                                       ;NO GO TO ERROR
1320 005114  104001                        ERROR   +1              ;CPU ERROR
1321 005116  012700  125252        2$:     MOV     #125252,R0              ;R0=125252
1322 005122  020027  125252                CMP     R0,#125252              ;DOES R0=125252
1323 005126  001401                        BEQ     3$                      ;YES GO ON
1324                                                                       ;NO GO TO ERROR
1325 005130  104001                        ERROR   +1              ;CPU ERROR
1326 005132  012700  052525        3$:     MOV     #52525,R0               ;R0=52525
```

J3

BASE INSTRUCTION SET TESTS

```
1327 005136  020027  052525                CMP     R0,#52525               ;DOES R0=52525
1328 005142  001401                         BEQ     4$                      ;YES GO ON
1329                                                                        ;NO GO TO ERROR
1330 005144  104001                         ERROR   +1              ;CPU ERROR
1331 005146                         4$:
1332                                ;
1335 005146                         GPR1TS:
1336                                ;       R1 BIT TESTS
1337 005146  012701  177777                MOV     #177777,R1              ;R1=177777
1338 005152  020127  177777                CMP     R1,#177777              ;DOES R1=177777
1339 005156  001401                         BEQ     1$                      ;YES GO ON
1340                                                                        ;NO GO TO ERROR
1341 005160  104001                         ERROR   +1              ;CPU ERROR
1342 005162  005001                 1$:     CLR     R1                      ;R1=0
1343 005164  020127  000000                CMP     R1,#0                   ;DOES R1=0
1344 005170  001401                         BEQ     2$                      ;YES GO ON
1345                                                                        ;NO GO TO ERROR
1346 005172  104001                         ERROR   +1              ;CPU ERROR
1347 005174  012701  125252         2$:     MOV     #125252,R1              ;R1=125252
1348 005200  020127  125252                CMP     R1,#125252              ;DOES R1=125252
1349 005204  001401                         BEQ     3$                      ;YES GO ON
1350                                                                        ;NO GO TO ERROR
1351 005206  104001                         ERROR   +1              ;CPU ERROR
1352 005210  012701  052525         3$:     MOV     #52525,R1               ;R1=52525
1353 005214  020127  052525                CMP     R1,#52525               ;DOES R1=52525
1354 005220  001401                         BEQ     4$                      ;YES GO ON
1355                                                                        ;NO GO TO ERROR
1356 005222  104001                         ERROR   +1              ;CPU ERROR
1357 005224                         4$:
1358                                ;
1361 005224                         GPR2TS:
1362                                ;       R2 BIT TESTS
1363 005224  012702  177777                MOV     #177777,R2              ;R2=177777
1364 005230  020227  177777                CMP     R2,#177777              ;DOES R2=177777
1365 005234  001401                         BEQ     1$                      ;YES GO ON
1366                                                                        ;NO GO TO ERROR
1367 005236  104001                         ERROR   +1              ;CPU ERROR
1368 005240  005002                 1$:     CLR     R2                      ;R2=0
1369 005242  020227  000000                CMP     R2,#0                   ;DOES R2=0
1370 005246  001401                         BEQ     2$                      ;YES GO ON
1371                                                                        ;NO GO TO ERROR
1372 005250  104001                         ERROR   +1              ;CPU ERROR
1373 005252  012702  125252         2$:     MOV     #125252,R2              ;R2=125252
1374 005256  020227  125252                CMP     R2,#125252              ;DOES R2=125252
1375 005262  001401                         BEQ     3$                      ;YES GO ON
1376                                                                        ;NO GO TO ERROR
1377 005264  104001                         ERROR   +1              ;CPU ERROR
1378 005266  012702  052525         3$:     MOV     #52525,R2               ;R2=52525
1379 005272  020227  052525                CMP     R2,#52525               ;DOES R2=52525
1380 005276  001401                         BEQ     4$                      ;YES GO ON
1381                                                                        ;NO GO TO ERROR
1382 005300  104001                         ERROR   +1              ;CPU ERROR
1383 005302                         4$:
1384                                ;
1387 005302                         GPR3TS:
1388                                ;       R3 BIT TESTS
1389 005302  012703  177777                MOV     #177777,R3              ;R3=177777
```

BASE INSTRUCTION SET TESTS

```
1390 005306  020327  177777            CMP     R3,#177777              ;DOES R3=177777
1391 005312  001401                    BEQ     1$                      ;YES GO ON
1392                                                                   ;NO GO TO ERROR
1393 005314  104001                    ERROR   +1              ;CPU ERROR
1394 005316  005003              1$:   CLR     R3                      ;R3=0
1395 005320  020327  000000            CMP     R3,#0                   ;DOES R3=0
1396 005324  001401                    BEQ     2$                      ;YES GO ON
1397                                                                   ;NO GO TO ERROR
1398 005326  104001                    ERROR   +1              ;CPU ERROR
1399 005330  012703  125252        2$: MOV     #125252,R3              ;R3=125252
1400 005334  020327  125252            CMP     R3,#125252              ;DOES R3=125252
1401 005340  001401                    BEQ     3$                      ;YES GO ON
1402                                                                   ;NO GO TO ERROR
1403 005342  104001                    ERROR   +1              ;CPU ERROR
1404 005344  012703  052525        3$: MOV     #52525,R3               ;R3=52525
1405 005350  020327  052525            CMP     R3,#52525               ;DOES R3=52525
1406 005354  001401                    BEQ     4$                      ;YES GO ON
1407                                                                   ;NO GO TO ERROR
1408 005356  104001                    ERROR   +1              ;CPU ERROR
1409 005360                      4$:
1410                                   ;
1413 005360                      GPR4TS:
1414                                   ;       R4 BIT TESTS
1415 005360  012704  177777            MOV     #177777,R4              ;R4=177777
1416 005364  020427  177777            CMP     R4,#177777              ;DOES R4=177777
1417 005370  001401                    BEQ     1$                      ;YES GO ON
1418                                                                   ;NO GO TO ERROR
1419 005372  104001                    ERROR   +1              ;CPU ERROR
1420 005374  005004              1$:   CLR     R4                      ;R4=0
1421 005376  020427  000000            CMP     R4,#0                   ;DOES R4=0
1422 005402  001401                    BEQ     2$                      ;YES GO ON
1423                                                                   ;NO GO TO ERROR
1424 005404  104001                    ERROR   +1              ;CPU ERROR
1425 005406  012704  125252        2$: MOV     #125252,R4              ;R4=125252
1426 005412  020427  125252            CMP     R4,#125252              ;DOES R4=125252
1427 005416  001401                    BEQ     3$                      ;YES GO ON
1428                                                                   ;NO GO TO ERROR
1429 005420  104001                    ERROR   +1              ;CPU ERROR
1430 005422  012704  052525        3$: MOV     #52525,R4               ;R4=52525
1431 005426  020427  052525            CMP     R4,#52525               ;DOES R4=52525
1432 005432  001401                    BEQ     4$                      ;YES GO ON
1433                                                                   ;NO GO TO ERROR
1434 005434  104001                    ERROR   +1              ;CPU ERROR
1435 005436                      4$:
1436                                   ;
1439 005436                      GPR5TS:
1440                                   ;       R5 BIT TESTS
1441 005436  012705  177777            MOV     #177777,R5              ;R5=177777
1442 005442  020527  177777            CMP     R5,#177777              ;DOES R5=177777
1443 005446  001401                    BEQ     1$                      ;YES GO ON
1444                                                                   ;NO GO TO ERROR
1445 005450  104001                    ERROR   +1              ;CPU ERROR
1446 005452  005005              1$:   CLR     R5                      ;R5=0
1447 005454  020527  000000            CMP     R5,#0                   ;DOES R5=0
1448 005460  001401                    BEQ     2$                      ;YES GO ON
1449                                                                   ;NO GO TO ERROR
1450 005462  104001                    ERROR   +1              ;CPU ERROR
```

L3

BASE INSTRUCTION SET TESTS

```
1451 005464  012705  125252          2$:     MOV     #125252,R5              ;R5=125252
1452 005470  020527  125252                  CMP     R5,#125252              ;DOES R5=125252
1453 005474  001401                          BEQ     3$                      ;YES GO ON
1454                                                                         ;NO GO TO ERROR
1455 005476  104001                          ERROR   .1              ;CPU ERROR
1456 005500  012705  052525          3$:     MOV     #52525,R5               ;R5=52525
1457 005504  020527  052525                  CMP     R5,#52525               ;DOES R5=52525
1458 005510  001401                          BEQ     4$                      ;YES GO ON
1459                                                                         ;NO GO TO ERROR
1460 005512  104001                          ERROR   .1              ;CPU ERROR
1461 005514                          4$:
1462                                 ;
1465 005514                          GPR6TS:
1466                                 ;       R6 BIT TESTS
1467 005514  012706  177777                  MOV     #177777,R6              ;R6=177777
1468 005520  020627  177777                  CMP     R6,#177777              ;DOES R6=177777
1469 005524  001401                          BEQ     1$                      ;YES GO ON
1470                                                                         ;NO GO TO ERROR
1471 005526  104001                          ERROR   .1              ;CPU ERROR
1472 005530  005006                  1$:     CLR     R6                      ;R6=0
1473 005532  020627  000000                  CMP     R6,#0                   ;DOES R6=0
1474 005536  001401                          BEQ     2$                      ;YES GO ON
1475                                                                         ;NO GO TO ERROR
1476 005540  104001                          ERROR   .1              ;CPU ERROR
1477 005542  012706  125252          2$:     MOV     #125252,R6              ;R6=125252
1478 005546  020627  125252                  CMP     R6,#125252              ;DOES R6=125252
1479 005552  001401                          BEQ     3$                      ;YES GO ON
1480                                                                         ;NO GO TO ERROR
1481 005554  104001                          ERROR   .1              ;CPU ERROR
1482 005556  012706  052525          3$:     MOV     #52525,R6               ;R6=52525
1483 005562  020627  052525                  CMP     R6,#52525               ;DOES R6=52525
1484 005566  001401                          BEQ     4$                      ;YES GO ON
1485                                                                         ;NO GO TO ERROR
1486 005570  104001                          ERROR   .1              ;CPU ERROR
1487 005572  012706  001000          4$:     MOV     #STBOT,R6               ;RESTORE SP
1488
1489                                 ;
1492 005576                          PSWBTS:
1493                                 ;       PSW LOW BYTE BIT TESTS
1494 005576  012737  000377  177776          MOV     #377,@#177776           ;PS=357 T BIT SHOULDN'T SET
1495 005604  022737  000357  177776          CMP     #357,@#177776           ;DOES PS=357
1496 005612  001401                          BEQ     1$                      ;YES GO ON
1497                                                                         ;NO GO TO ERROR
1498 005614  104001                          ERROR   .1              ;CPU ERROR
1499 005616  005037  177776          1$:     CLR     @#177776                ;PS=0
1500 005622  022737  000000  177776          CMP     #0,@#177776             ;DOES PS=0
1501 005630  001401                          BEQ     2$                      ;YES GO ON
1502                                                                         ;NO GO TO ERROR
1503 005632  104001                          ERROR   .1              ;CPU ERROR
1504 005634  012737  000105  177776  2$:     MOV     #105,@#177776           ;PS=105
1505 005642  022737  000105  177776          CMP     #105,@#177776           ;DOES PS=105
1506 005650  001401                          BEQ     3$                      ;YES GO ON
1507                                                                         ;NO GO TO ERROR
1508 005652  104001                          ERROR   .1              ;CPU ERROR
1509 005654  012737  000252  177776  3$:     MOV     #252,@#177776           ;PS=252
1510 005662  022737  000252  177776          CMP     #252,@#177776           ;DOES PS=252
1511 005670  001401                          BEQ     4$                      ;YES GO ON
```

```
                                                                        ;NO GO TO ERROR
005672  104001                          ERROR   +1              ;CPU ERROR
005674                          4$:

005674                          MSPO:

                                ;       TEST SINGLE OPERAND INSTRUCTIONS- MODE 0
                                ;;*****************************************************************
                                ;THE INC, COM, CLR, AND DECREMENT INSTRUCTIONS ARE VERIFED.
005674  005004                          CLR     R4                      ;INITIALIZE R4 WITH DATA
005676  005104                          COM     R4                      ;
005700  005004                          CLR     R4                      ;*TEST INSTRUCTION
005702  001401                          BEQ     2$                      ;BRANCH IF R4 CLEARED
005704  104001                  1$:     ERROR   +1              ;CPU ERROR
005706  005104                  2$:     COM     R4                      ;*TEST COMPLIMENT INSTRUCTION
005710  005204                          INC     R4                      ;*TEST INCREMENT INSTRUCTION
005712  001401                          BEQ     4$                      ;BRANCH IF R4 =0
                                                                        ;COMPLIMENT OR INCREMENT FAILED
005714  104001                  3$:     ERROR   +1              ;CPU ERROR
005716                          4$:

                                ;
005716                          MSPB:

                                ;       TEST SINGLE OPS - EVEN BYTE OF CLRB, DECB, AND COMB
005716  005004                          CLR     R4
005720  005104                          COM     R4              ;SETUP TEST REGISTER
005722  105004                          CLRB    R4              ;*TEST CLEAR BYTE INSTRUCTION
005724  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;CLEAR EVEN BYTE FAILED
005726  104001                  1$:     ERROR   +1              ;CPU ERROR
005730  105304                  2$:     DECB    R4              ;*TEST DECREMENT BYTE
005732  100002                          BPL     3$              ;DECREMENT BYTE FAILED
005734  105104                          COMB    R4              ;*TEST COMPLIMENT BYTE
005736  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;COMPLIMENT OR DECREMENT FAILED TO WORK
005740  104001                  3$:     ERROR   +1              ;CPU ERROR
005742                          4$:

                                ;
005742                          MSPC:
                                ;       TEST SINGLE OPS - MODE 1 CLRB, COMB, AND INCB
005742  005004                          CLR     R4
005744  005014                          CLR     (R4)
005746  005114                          COM     (R4)            ;SETUP TEST DATA
005750  005014                          CLR     (R4)            ;*TEST INSTRUCTION
005752  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;MODE 1 FAILED
005754  104001                  1$:     ERROR   +1              ;CPU ERROR
005756  005114                  2$:     COM     (R4)            ;*TEST INSTRUCTION
005760  001403                          BEQ     3$              ;(0)SHOULD = -1
005762  100002                          BPL     3$              ;
005764  005214                          INC     (R4)            ;*TEST INSTRUCTION
005766  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;COM OR INC FAILED TO ALTER LOC 0 CORRECTLY
005770  104001                  3$:     ERROR   +1              ;CPU ERROR
005772                          4$:
```

BASE INSTRUCTION SET TESTS

```
1581
1582                                    ;
1585 005772                             MSPD:
1586
1587                                    ;       TEST SINGLE OPS MODE1-EVEN BYTE-CLRB,COMB,INCB
1588 005772  005004                             CLR     R4
1589 005774  005014                             CLR     (R4)
1590 005776  005114                             COM     (R4)            ;SETUP TEST DATA
1591 006000  105014                             CLRB    (R4)            ;*TEST INSTRUCTION
1592 006002  105014                             CLRB    (R4)            ;*TEST INSTRUCTION
1593 006004  001401                             BEQ     2$              ;BRANCH IF GOOD
1594                                                                     ;CLEAR (0) EVEN BYTE FAILED
1595 006006  104001                     1$:     ERROR   +1              ;CPU ERROR
1596 006010  105214                     2$:     INCB    (R4)            ;*TEST INSTRUCTION
1597 006012  100405                             BMI     3$              ; TEST FLAGS
1598 006014  001404                             BEQ     3$
1599 006016  105114                             COMB    (R4)            ;*TEST INSTRUCTION
1600 006020  105214                             INCB    (R4)
1601 006022  105214                             INCB    (R4)
1602 006024  001401                             BEQ     4$              ;BRANCH IF GOOD
1603                                                                     ;COMB OR INCB FAILED
1604 006026  104001                     3$:     ERROR   +1              ;CPU ERROR
1605 006030                             4$:
1606
1607                                    ;
1610 006030                             MSPEO:
1611
1612                                    ;       TEST SINGLE OPS - ODD BYTE - CLRB, COMB, DECB
1613 006030  005004                             CLR     R4
1614 006032  005014                             CLR     (R4)
1615 006034  005114                             COM     (R4)            ;SETUP TEST DATA
1616 006036  005204                             INC     R4              ;POINT TO ODD BYTE
1617 006040  105014                             CLRB    (R4)            ;*TEST INSTRUCTION
1618 006042  001401                             BEQ     1$              ;BRANCH IF GOOD
1619                                                                     ;CLEAR ODD BYTE FAILED
1620 006044  104001                             ERROR   +1              ;CPU ERROR
1621 006046  005304                     1$:     DEC     R4              ;POINT TO EVEN BYTE
1622 006050  005214                             INC     (R4)            ;LOC 0=1 0
1623 006052  005204                             INC     R4              ;POINT TO ODD BYTE
1624 006054  105114                             COMB    (R4)            ;*TEST INSTRUCTION
1625 006056  105214                             INCB    (R4)            ;LOC 0=-1 0
1626 006060  100003                             BPL     2$              ;BRANCH IF ERROR
1627 006062  001402                             BEQ     2$              ;
1628 006064  105214                             INCB    (R4)            ;*TEST INSTRUCTION
1629 006066  001401                             BEQ     3$              ;BRANCH IF GOOD
1630                                                                     ;MODE 1, ODD BYTE FAILED
1631 006070  104001                     2$:     ERROR   +1              ;CPU ERROR
1632 006072                             3$:
1633
1634                                    ;
1637 006072                             MSPF:
1638                                    ;       TEST SINGLE OP - MODE 2 - CLR, COM, INC
1639 006072  005004                             CLR     R4
1640 006074  105104                             COMB    R4
1641 006076  005204                             INC     R4              ;R4=400
1642 006100  005014                             CLR     (R4)            ;400=0
1643 006102  005114                             COM     (R4)            ;400=-1
```

BASE INSTRUCTION SET TESTS

```
1644 006104  005024                    CLR     (R4)+           ;*TEST INSTRUCTION
1645 006106  001401                    BEQ     1$              ;BRANCH IF GOOD
1646                                                           ;MODE 2 CLEAR FAILED
1647 006110  104001                    ERROR   ^1              ;CPU ERROR
1648 006112  005304            1$:     DEC     R4
1649 006114  005304                    DEC     R4              ;R4=400
1650 006116  005124                    COM     (R4)+           ;*TEST INSTRUCTION
1651 006120  100004                    BPL     2$              ;BRANCH IF FAILURE
1652 006122  005304                    DEC     R4
1653 006124  005304                    DEC     R4              ;R4=400
1654 006126  005224                    INC     (R4)+           ;*TEST INSTRUCTION
1655 006130  001401                    BEQ     3$              ;BRANCH IF GOOD
1656                                                           ;MODE 2 FAILURE
1657 006132  104001            2$:     ERROR   ^1              ;CPU ERROR
1658 006134                    3$:
1659
1660                           ;
1663 006134                    MSPG:
1664
1665                           ;       TEST CLRB, COMB, DECB, MODE 2 - EVEN BYTE
1666 006134  005004                    CLR     R4
1667 006136  105104                    COMB    R4
1668 006140  005204                    INC     R4              ;R4=400
1669 006142  005014                    CLR     (R4)
1670 006144  005114                    COM     (R4)            ;400=-1
1671 006146  105024                    CLRB    (R4)+           ;*TEST INSTRUCTION
1672 006150  001401                    BEQ     1$              ;BRANCH IF GOOD
1673                                                           ;MODE 2 EVEN BYTE FAILED
1674 006152  104001                    ERROR   ^1              ;CPU ERROR
1675 006154  005304            1$:     DEC     R4
1676 006156  105324                    DECB    (R4)+           ;*TEST INSTRUCTION
1677 006160  100003                    BPL     2$              ;BRANCH IF BAD
1678 006162  005304                    DEC     R4              ;POINT TO EVEN BYTE
1679 006164  105124                    COMB    (R4)+           ;*TEST INSTRUCTION
1680 006166  001401                    BEQ     3$              ;BRANCH IF GOOD
1681                                                           ;MODE 2, EVEN BYTE FAILED
1682 006170  104001            2$:     ERROR   ^1              ;CPU ERROR
1683 006172                    3$:
1684
1685                           ;
1688 006172                    MSPH:
1689
1690                           ;       TEST CLRB, COMB, INCB MODE 2 - ODD BYTE
1691 006172  005004                    CLR     R4
1692 006174  105104                    COMB    R4
1693 006176  005204                    INC     R4              ;R4=400
1694 006200  005014                    CLR     (R4)
1695 006202  005114                    COM     (R4)            ;400=-1 -1
1696 006204  005214                    INC     (R4)            ;POINT TO ODD BYTE
1697 006206  105024                    CLRB    (R4)+           ;*TEST INSTRUCTION
1698 006210  001401                    BEQ     1$              ;BRANCH IF GOOD
1699                                                           ;MODE 2,ODD BYTE FAILED
1700 006212  104001                    ERROR   ^1              ;CPU ERROR
1701 006214  005304            1$:     DEC     R4              ;POINT TO ODD BYTE
1702 006216  005304                    DEC     R4
1703 006220  105224                    INCB    (R4)+           ;400=1 0
1704 006222  105124                    COMB    (R4)+           ;*TEST INSTRUCTION
```

```
1705 006224  100003                    BPL     2$              ;BRANCH IF MODE 2 FAILED
1706 006226  005304                    DEC     R4              ;POINT TO ODD BYTE
1707 006230  105224                    INCB    (R4)+
1708 006232  001401                    BEQ     3$              ;BRANCH IF GOOD
1709                                                           ;MODE 2,ODD BYTE FAILED
1710 006234  104001              2$:   ERROR   .1              ;CPU ERROR
1711 006236                      3$:
1712
1713                             ;
1716 006236                      MSPI:
1717
1718                             ;       TEST CLR, COM, INC - MODE 3
1719 006236  005004                    CLR     R4              ;
1720 006240  005014                    CLR     (R4)            ;0-0
1721 006242  105114                    COMB    (R4)            ;
1722 006244  005214                    INC     (R4)            ;0-400
1723 006246  005034                    CLR     @(R4)+          ;*TEST INSTRUCTION
1724 006250  001401                    BEQ     1$              ;BRANCH IF GOOD
1725                                                           ;MODE 3 FAILED, 400 SHOULD-0
1726 006252  104001                    ERROR   .1              ;CPU ERROR
1727 006254  005304              1$:   DEC     R4
1728 006256  005304                    DEC     R4              ;R4-0
1729 006260  005134                    COM     @(R4)+          ;*TEST INSTRUCTION
1730 006262  100004                    BPL     2$              ;BRANCH IF BAD
1731 006264  005304                    DEC     R4
1732 006266  005304                    DEC     R4              ;REPOSITION POINTER
1733 006270  005234                    INC     @(R4)+          ;*TEST INSTRUCTION
1734 006272  001401                    BEQ     3$              ;BRANCH IF GOOD
1735                                                           ;MODE 3 FAILED
1736 006274  104001              2$:   ERROR   .1              ;CPU ERROR
1737 006276                      3$:
1738
1739                             ;
1742 006276                      MSPJ:
1743
1744                             ;       TEST CLRB, COMB, INCB - MODE 3, EVEN/ODD BYTE
1745 006276  005004                    CLR     R4              ;R4-0
1746 006300  005001                    CLR     R1
1747 006302  105101                    COMB    R1
1748 006304  005201                    INC     R1              ;R1-400
1749 006306  005011                    CLR     (R1)
1750 006310  005121                    COM     (R1)+           ;400--1
1751 006312  005011                    CLR     (R1)            ;
1752 006314  105111                    COMB    (R1)            ;402-000 377
1753 006316  005014                    CLR     (R4)            ;
1754 006320  105114                    COMB    (R4)            ;
1755 006322  005214                    INC     (R4)            ;0-400
1756 006324  105034                    CLRB    @(R4)+          ;*TEST INSTRUCTION 400-377 000
1757 006326  001401                    BEQ     1$              ;BRANCH IF MODE 3 EVEN BYTE CLEARED
1758                                                           ; TEST INSTRUCTION FAILED
1759 006330  104001                    ERROR   .1              ;CPU ERROR
1760 006332  005304              1$:   DEC     R4              ;REPOSTION POINTER
1761 006334  005304                    DEC     R4
1762 006336  105134                    COMB    @(R4)+          ;*TEST INSTRUCTION
1763 006340  005304                    DEC     R4
1764 006342  005304                    DEC     R4              ;REPOSITION POINTER
1765 006344  105234                    INCB    @(R4)+          ;*TEST INSTRUCTION
```

```
1766 006346  001401                    BEQ     3$              ;BRANCH IF GOOD
1767                                                           ;MODE 3, EVEN BYTE FAILED
1768 006350  104001            2$:     ERROR   +1              ;CPU ERROR
1769 006352  005304            3$:     DEC     R4
1770 006354  005304                    DEC     R4
1771 006356  005214                    INC     (R4)            ;R4=401
1772 006360  105234                    INCB    @(R4)+          ;*TEST INSTRUCTION
1773 006362  001004                    BNE     4$              ;BRANCH IF 402 NEQ 0
1774 006364  005304                    DEC     R4
1775 006366  005304                    DEC     R4              ;R4=401
1776 006370  105034                    CLRB    @(R4)+          ;401=0
1777 006372  001401                    BEQ     5$              ;BRANCH IF GOOD
1778                                                           ;ODD BYTE FAILED
1779 006374  104001            4$:     ERROR   +1              ;CPU ERROR
1780 006376  005304            5$:     DEC     R4
1781 006400  005304                    DEC     R4              ;R4=401
1782 006402  105134                    COMB    @(R4)+          ;403=377
1783 006404  005304                    DEC     R4
1784 006406  005304                    DEC     R4
1785 006410  105234                    INCB    @(R4)+          ;*TEST INSTRUCTION
1786 006412  001401                    BEQ     7$              ;BRANCH IF GOOD
1787                                                           ;MODE3 ODD BYTE FAILED.
1788 006414  104001            6$:     ERROR   +1              ;CPU ERROR
1789 006416                    7$:
1790
1791                            ;
1794 006416                    MSPL:
1795
1796                            ;       TEST CLR, COM, DEC - MODE 4
1797 006416  005004                    CLR     R4
1798 006420  105104                    COMB    R4
1799 006422  005204                    INC     R4              ;R4=400
1800 006424  005014                    CLR     (R4)            ;
1801 006426  005124                    COM     (R4)+           ;400=-1
1802 006430  005014                    CLR     (R4)            ;
1803 006432  005224                    INC     (R4)+           ;402=1
1804 006434  005044                    CLR     -(R4)           ;*TEST INSTRUCTION
1805 006436  001401                    BEQ     1$              ;BRANCH IF GOOD
1806                                                           ;MODE 4 FAILED
1807 006440  104001                    ERROR   +1              ;CPU ERROR
1808 006442  005344            1$:     DEC     -(R4)           ;*TEST INSTRUCTION 400=-2
1809 006444  005114                    COM     (R4)            ;400=1
1810 006446  001405                    BEQ     2$              ;BRANCH IF BAD
1811 006450  100404                    BMI     2$              ;BRANCH IF BAD
1812 006452  005204                    INC     R4
1813 006454  005204                    INC     R4              ;R4=400
1814 006456  005344                    DEC     -(R4)           ;*TEST INSTRUCTION
1815 006460  001401                    BEQ     3$              ;BRANCH IF GOOD
1816                                                           ;MODE 4 FAILED
1817 006462  104001            2$:     ERROR   +1              ;CPU ERROR
1818 006464                    3$:
1819
1820                            ;
1823 006464                    MSPM:
1824
1825                            ;       TEST COMB, INCB, CLRB - MODE 4, ODD BYTE
1826 006464  005004                    CLR     R4
```

BASE INSTRUCTION SET TESTS

```
1827 006466  105104                    COMB    R4              ;
1828 006470  005204                    INC     R4              ;R4=400
1829 006472  005044                    CLR     -(R4)           ;376=0
1830 006474  105114                    COMB    (R4)
1831 006476  005224                    INC     (R4)+           ;376=001 000
1832 006500  005014                    CLR     (R4)
1833 006502  005124                    COM     (R4)+           ;400=1
1834 006504  005204                    INC     R4              ;R4=403
1835 006506  105044                    CLRB    -(R4)           ; TEST INST. CLEAR ODD BYTE (401)
1836 006510  001401                    BEQ     2$              ;BRANCH IF GOOD
1837                                                           ;MODE 4 BYTE FAILED
1838 006512  104001              1$:   ERROR   +1              ;CPU ERROR
1839 006514  005204              2$:   INC     R4
1840 006516  005204                    INC     R4              ;R4=403
1841 006520  105144                    COMB    -(R4)           ; TEST INST. 401=377
1842 006522  005304                    DEC     R4
1843 006524  005304                    DEC     R4
1844 006526  105244                    INCB    -(R4)           ; TEST INST. 401=0
1845 006530  001401                    BEQ     4$              ;BRANCH IF GOOD
1846                                                           ;MODE 4 ODD BYTE FAILED
1847 006532  104001              3$:   ERROR   +1              ;CPU ERROR
1848 006534  105344              4$:   DECB    -(R4)           ;*TEST INST.
1849 006536  001401                    BEQ     6$              ;BRANCH IF GOOD
1850                                                           ;MODE 4 DECREMENT OCD BYTE FAILED
1851 006540  104001              5$:   ERROR   +1              ;CPU ERROR
1852 006542                      6$:
1853
1854                             ;
1857 006542                      MSPN:
1858
1859                             ;       TEST CLR, COM, INC - MODE 5
1860 006542  005004                    CLR     R4
1861 006544  005014                    CLR     (R4)
1862 006546  105114                    COMB    (R4)
1863 006550  005224                    INC     (R4)+           ;0=400
1864 006552  005054                    CLR     @-(R4)          ;*TEST INST. 400=0
1865 006554  001401                    BEQ     1$              ;BRANCH IF GOOD
1866                                                           ;MODE 5 FAILED
1867 006556  104001                    ERROR   +1              ;CPU ERROR
1868 006560  005204              1$:   INC     R4
1869 006562  005204                    INC     R4              ;RESET POINTER TO 0
1870 006564  005154                    COM     @-(R4)          ;*TEST INST. 376=1
1871 006566  001407                    BEQ     2$              ;BRANCH IF BAD
1872 006570  005204                    INC     R4
1873 006572  005204                    INC     R4              ;REPOSITION POINTER
1874 006574  005354                    DEC     @-(R4)          ;*TEST INST. 376=0
1875 006576  001403                    BEQ     2$              ;BRANCH IF BAD
1876 006600  005224                    INC     (R4)+           ;0=401 R4=2
1877 006602  105254                    INCB    @-(R4)          ;*TEST INST.,400= 0 376
1878 006604  001401                    BEQ     3$              ;BRANCH IF GOOD
1879                                                           ;MODE 5 FAILED
1880 006606  104001              2$:   ERROR   +1              ;CPU ERROR
1881 006610                      3$:
1882
1883                             ;
1886 006610                      MSP0:
1887
```

BASE INSTRUCTION SET TESTS

```
1888                                    ;       TEST NEG MODE 5
1889 006610  005004                             CLR     R4
1890 006612  105104                             COMB    R4
1891 006614  005204                             INC     R4              ;R4=400
1892 006616  005001                             CLR     R1
1893 006620  105101                             COMB    R1
1894 006622  005301                             DEC     R1              ;R1=376
1895 006624  005002                             CLR     R2              ;R2=0
1896 006626  005012                             CLR     (R2)            ;0=0
1897 006630  005014                             CLR     (R4)            ;
1898 006632  005114                             COM     (R4)            ;400=-1
1899 006634  005011                             CLR     (R1)            ;376=0
1900 006636  005454                             NEG     @-(R4)          ;0=0
1901 006640  001401                             BEQ     2$              ;BRANCH IF GOOD
1902                                                                     ;NEG FAILED
1903 006642  104001                     1$:     ERROR   +1              ;CPU ERROR
1904 006644  005334                     2$:     DEC     @(R4)+          ;0=-1
1905 006646  005454                             NEG     @-(R4)          ;0=1
1906 006650  001403                             BEQ     3$              ;BRANCH IF BAD
1907 006652  102402                             BVS     3$              ;BRANCH IF BAD
1908 006654  100401                             BMI     3$              ;BRANCH IF BAD
1909 006656  103401                             BCS     4$              ;BRANCH IF GOOD
1910                                                                     ;NEG FAILED
1911 006660  104001                     3$:     ERROR   +1              ;CPU ERROR
1912 006662  005334                     4$:     DEC     @(R4)+          ; TEST RESULT OF NEGATE
1913 006664  001401                             BEQ     6$              ;BRANCH IF GOOD
1914                                                                     ;RESULT OF NEGATE BAD
1915 006666  104001                     5$:     ERROR   +1              ;CPU ERROR
1916 006670  105212                     6$:     INCB    (R2)            ;0=1
1917 006672  005454                             NEG     @-(R4)          ;0=-1
1918 006674  001403                             BEQ     7$              ;
1919 006676  102402                             BVS     7$
1920 006700  103001                             BCC     7$              ;
1921 006702  100401                             BMI     8$              ;BRANCH IF GOOD
1922                                                                     ;BAD NEGATE
1923 006704  104001                     7$:     ERROR   +1              ;CPU ERROR
1924 006706  105212                     8$:     INCB    (R2)            ;0=0
1925 006710  001401                             BEQ     10$             ;BRANCH IF GOOD
1926 006712  104001                     9$:     ERROR   +1              ;CPU ERROR
1927 006714                             10$:
1928
1929                                    ;
1931
1933 006714                             MSPP:
1934
1935                                    ;       TEST CLR, COM, INC - MODE 6
1936 006714  005004                             CLR     R4
1937 006716  005204                             INC     R4
1938 006720  005204                             INC     R4              ;R4=2
1939 006722  005001                             CLR     R1
1940 006724  105101                             COMB    R1
1941 006726  005201                             INC     R1              ;R1=400
1942 006730  005011                             CLR     (R1)            ;
1943 006732  005121                             COM     (R1)+           ;400=-1
1944 006734  005011                             CLR     (R1)            ;R1=402
1945 006736  005211                             INC     (R1)            ;402=1
1946 006740  005002                             CLR     R2              ;R2=0
```

BASE INSTRUCTION SET TESTS

```
1947 006742  005012                        CLR     (R2)            ;0=0
1948 006744  005064  000376                CLR     376(R4)         ;400=0
1949 006750  001401                        BEQ     2$              ;BRANCH IF GOOD
1950 006752  104001                1$:     ERROR   +1              ;CPU ERROR
1951 006754  005364  000376        2$:     DEC     376(R4)         ;400=-1
1952 006760  005164  000400                COM     400(R4)         ;402=-1
1953 006764  001405                        BEQ     3$              ;BRANCH IF BAD
1954 006766  005264  000400                INC     400(R4)         ;402=-2
1955 006772  005264  000400                INC     400(R4)
1956 006776  001401                        BEQ     4$              ;BRANCH IF GOOD
1957                                                               ;MODE 6 FAILED
1958 007000  104001                3$:     ERROR   +1              ;CPU ERROR
1959 007002  005261  177776        4$:     INC     -2(R1)          ;400=0
1960 007006  001401                        BEQ     6$              ;BRANCH IF GOOD
1961                                                               ;INC MODE 6 FAILED
1962 007010  104001                5$:     ERROR   +1              ;CPU ERROR
1963 007012                        6$:
1964
1965                               ;
1967                               ;
1968                               ;
1970 007012                        MSPQ:
1971
1972                               ;       TEST NEG MODE 6
1973 007012  005001                        CLR     R1              ;R1=0
1974 007014  005004                        CLR     R4
1975 007016  105104                        COMB    R4
1976 007020  005204                        INC     R4              ;R4=400
1977 007022  005014                        CLR     (R4)            ;
1978 007024  005114                        COM     (R4)            ;400=-1
1979 007026  005044                        CLR     -(R4)           ;376=0
1980 007030  005044                        CLR     -(R4)           ;
1981 007032  005224                        INC     (R4)+           ;374=1 R4=376
1982 007034  005464  000002                NEG     2(R4)           ;400=1
1983 007040  001403                        BEQ     1$              ;NEGATE FAILED
1984 007042  102402                        BVS     1$
1985 007044  100401                        BMI     1$
1986 007046  103401                        BCS     2$              ;BRANCH IF GOOD
1987                                                               ;NEGATE FAILED
1988 007050  104001                1$:     ERROR   +1              ;CPU ERROR
1989 007052  005364  000002        2$:     DEC     2(R4)           ; TEST RESULT OF NEGATE
1990 007056  001401                        BEQ     4$              ;BRANCH IF GOOD
1991                                                               ;RESULT OF NEGATE FAILED
1992 007060  104001                3$:     ERROR   +1              ;CPU ERROR
1993 007062  005464  000000        4$:     NEG     0(R4)           ;+0=0
1994 007066  001401                        BEQ     5$              ; BRANCH IF GOOD
1995                                                               ;NEGATE FAILED
1996 007070  104001                        ERROR   +1              ;CPU ERROR
1997 007072  105461  000374        5$:     NEGB    374(R1)         ;374=0 377
1998 007076  102403                        BVS     6$              ;
1999 007100  001402                        BEQ     6$
2000 007102  100001                        BPL     6$
2001 007104  103401                        BCS     7$              ;BRANCH IF GOOD
2002                                                               ;NEGATE FAILED
2003 007106  104001                6$:     ERROR   +1              ;CPU ERROR
2004 007110  105261  000374        7$:     INCB    374(R1)         ;374=0
2005 007114  001401                        BEQ     9$              ;BRANCH IF GOOD
```

```
                                                                        ;NEGATE FAILED
007116  104001                  8$:     ERROR   +1                      ;CPU ERROR
007120                          9$:

                                ;
007120                          MSPR:

                                ;       TEST CLR, COM, INC - MODE 7
007120  005001                          CLR     R1                      ;R1=0
007122  005004                          CLR     R4                      ;
007124  105104                          COMB    R4                      ;
007126  005204                          INC     R4                      ;R4=400
007130  005011                          CLR     (R1)
007132  105111                          COMB    (R1)
007134  005211                          INC     (R1)
007136  005211                          INC     (R1)
007140  005211                          INC     (R1)                    ;;0=402
007142  005014                          CLR     (R4)                    ;400=0
007144  005064  000002                  CLR     2(R4)                   ;
007150  005164  000002                  COM     2(R4)                   ;402=-1
007154  005074  177400                  CLR     @-400(R4)               ;402=0
007160  001401                          BEQ     2$                      ;BRANCH IF GOOD
007162  104001                  1$:     ERROR   +1                      ;CPU ERROR
                                                                        ;INSTRUCTION FAILED
007164  005171  000000          2$:     COM     @0(R1)                  ;402=-1
007170  100401                          BMI     4$                      ;BRANCH IF GOOD
007172  104001                  3$:     ERROR   +1                      ;CPU ERROR

007174  005104                  4$:     COM     R4
007176  005274  000401                  INC     @401(R4)                        ;402=0
007202  001401                          BEQ     6$                      ;BRANCH IF GOOD
007204  104001                  5$:     ERROR   +1                      ;CPU ERROR
                                                                        ;MODE 7 FAILED
007206                          6$:

                                ;
007206                          MSPS:

                                ;       TEST NEG MODE 7
007206  005004                          CLR     R4
007210  005014                          CLR     (R4)                    ;0=0
007212  005002                          CLR     R2                      ;
007214  105102                          COMB    R2
007216  005202                          INC     R2                      ;R2=400
007220  005012                          CLR     (R2)                    ;400=0
007222  005472  177400                  NEG     @-400(R2)                       ;NEG OF 0=0
007226  103401                          BCS     1$                      ;****
007230  001401                          BEQ     2$                      ;BRANCH IF GOOD
007232  104001                  1$:     ERROR   +1                      ;CPU ERROR
                                                                        ;
007234  005314                  2$:     DEC     (R4)                    ;0=-1
007236  005474  000400                  NEG     @+400(R4)                       ;0=1
007242  001403                          BEQ     3$                      ;BRANCH IF ERROR
007244  102402                          BVS     3$                      ;
007246  100401                          BMI     3$                      ;
007250  103401                          BCS     4$                      ;BRANCH IF GOOD
007252  104001                  3$:     ERROR   +1                      ;CPU ERROR
```

```
                                                        ;NEGATE MODE 7 FAILED
007254                          4$:
                                ;
                                MSPT:
                                ;       TEST SINGLE OPERAND MODE 2 REG 7
007254  005004                          CLR     R4
007256  105104                          COMB    R4
007260  005204                          INC     R4              ;R4=400
007262  005027                          CLR     (R7)+           ;CLEAR NEXT LOCATION
007264  177777                  1$:     .WORD   -1              ;SETUP INITIAL DATA
007266  001401                          BEQ     3$              ;BRANCH IF GOOD
007270  104001                  2$:     ERROR   +1              ;CPU ERROR
007272                          3$:
                                ;
                                MSPU:
                                ;       TEST TST MODE 0
007272  005004                          CLR     R4              ;R4=0
007274  000277                          SCC                     ;CONDITION CODES =1111
007276  000244                          CLZ                     ;CC=1011
007300  005704                          TST     R4              ;+TEST INSTRUCTION
007302  103403                          BCS     1$
007304  102402                          BVS     1$
007306  100401                          BMI     1$              ;BRANCH IF ERROR
007310  001401                          BEQ     2$              ;BRANCH IF GOOD
007312  104001                  1$:     ERROR   +1              ;CPU ERROR
                                                                ;TST MODE 0 FAILED
007314  005304                  2$:     DEC     R4              ;R4=-1
007316  000277                          SCC
007320  000250                          CLN                     ;CC=0111
007322  005704                          TST     R4              ;+TEST INSTRUCTION MODE 0
007324  103403                          BCS     3$              ;BRANCH IF ERROR
007326  102402                          BVS     3$
007330  001401                          BEQ     3$
007332  100401                          BMI     4$              ;BRANCH IF GOOD
007334  104001                  3$:     ERROR   +1              ;CPU ERROR
                                                                ;TST FAILED
007336                          4$:
                                ;
                                MSPVO:
                                ;       TEST TST MODE 0 BYTE
007336  005004                          CLR     R4
007340  105104                          COMB    R4              ;0=000 377
007342  000277                          SCC
007344  000250                          CLN                     ;CC=0111
007346  105704                          TSTB    R4              ;+TEST INSTRUCTION ON EVEN BYTE
007350  102403                          BVS     1$              ;BRANCH IF ERROR
007352  103402                          BCS     1$              ;
007354  102401                          BVS     1$              ;
007356  100401                          BMI     2$              ;BRANCH IF GOOD
007360  104001                  1$:     ERROR   +1              ;CPU ERROR
```

```
2130
2131 007362  005204                2$:     INC     R4              ;
2132 007364  105704                        TSTB    R4              ;POINT TO 1
2133 007366  001401                        BEQ     4$              ; TEST INSTRUCTION
2134 007370  104001                3$:     ERROR   +1              ;BRANCH IF GOOD
2135                                                               ;CPU ERROR
2136 007372                        4$:                             ;TST FAILED ON BYTE
2137
2138                               ;
2141 007372                        MSPV:
2142
2143                               ;       TEST TST MODE 1
2144 007372  005004                        CLR     R4
2145 007374  005014                        CLR     (R4)            ;0=0
2146 007376  000277                        SCC
2147 007400  000244                        CLZ                     ;CC=1011
2148 007402  005714                        TST     (R4)            ;*TEST INSTRUCTION IN MODE 1
2149 007404  103403                        BCS     1$              ;BRANCH IF ERROR
2150 007406  102402                        BVS     1$
2151 007410  100401                        BMI     1$              ;
2152 007412  001401                        BEQ     2$              ;BRANCH IF GOOD
2153 007414  104001                1$:     ERROR   +1              ;CPU ERROR
2154
2155 007416  005214                2$:     INC     (R4)            ;0=1
2156 007420  000277                        SCC
2157 007422  005714                        TST     (R4)            ; TEST INSTRUCTION
2158 007424  001403                        BEQ     3$              ;BRANCH IF ERROR
2159 007426  102402                        BVS     3$
2160 007430  103401                        BCS     3$
2161 007432  100001                        BPL     4$              ;BRANCH IF GOOD
2162 007434  104001                3$:     ERROR   +1              ;CPU ERROR
2163                                                               ;TST FAILED MODE 1
2164 007436                        4$:
2165
2166                               ;
2169 007436                        MSPX:
2170
2171                               ;       TEST TST MODE 1 BYTE
2172 007436  005004                        CLR     R4              ;R4=0
2173 007440  005014                        CLR     (R4)
2174 007442  105114                        COMB    (R4)
2175 007444  005214                        INC     (R4)            ;0=001 000
2176 007446  000277                        SCC
2177 007450  000244                        CLZ                     ;CC=1011
2178 007452  105714                        TSTB    (R4)            ;*TEST INTRUCTION
2179 007454  103403                        BCS     1$              ;BRANCH IF ERROR
2180 007456  102402                        BVS     1$              ;
2181 007460  100401                        BMI     1$
2182 007462  001401                        BEQ     2$              ;BRANCH IF GOOD
2183 007464  104001                1$:     ERROR   +1              ;CPU ERROR
2184
2185 007466  005204                2$:     INC     R4              ;R4=1
2186 007470  000277                        SCC
2187 007472  105714                        TSTB    (R4)            ; TEST INSTRUCTION
2188 007474  001403                        BEQ     3$              ;BRANCH IF ERROR
2189 007476  100402                        BMI     3$
2190 007500  102401                        BVS     3$
```

BASE INSTRUCTION SET TESTS

```
2191 007502  103001                    BCC     4$              ;BRANCH IF GOOD
2192 007504  104001            3$:     ERROR   +1              ;CPU ERROR
2193                                                           ;
2194 007506                    4$:
2195
2196                           ;
2199 007506                    MSPY:
2200
2201                           ;       TEST TST MODE 2
2202 007506  005004                    CLR     R4              ;
2203 007510  005024                    CLR     (R4)+           ;0=0
2204 007512  005014                    CLR     (R4)
2205 007514  005114                    COM     (R4)            ;2=-1
2206 007516  005004                    CLR     R4              ;R4=0
2207 007520  000277                    SCC             ;
2208 007522  000244                    CLZ             ;CC=1011
2209 007524  005724                    TST     (R4)+           ; TEST INSTRUCTION
2210 007526  103403                    BCS     1$              ;BRANCH IF ERROR
2211 007530  102402                    BVS     1$
2212 007532  100401                    BMI     1$
2213 007534  001401                    BEQ     2$              ;BRANCH IF GOOD
2214 007536  104001            1$:     ERROR   +1              ;CPU ERROR
2215                                                           ;MODE 2 TEST FAILED
2216 007540  005724            2$:     TST     (R4)+           ;TST LOC2
2217 007542  103403                    BCS     3$
2218 007544  102402                    BVS     3$
2219 007546  001401                    BEQ     3$
2220 007550  100401                    BMI     4$
2221 007552  104001            3$:     ERROR   +1              ;CPU ERROR
2222                                                           ;MODE 2 FAILED
2223 007554                    4$:
2224
2225                           ;
2228 007554                    MSPZ:
2229
2230                           ;       TEST TST MODE 2 BYTE
2231 007554  005004                    CLR     R4
2232 007556  005024                    CLR     (R4)+           ;
2233 007560  105144                    COMB    -(R4)           ;0=377 000
2234 007562  005304                    DEC     R4              ;R4=0
2235 007564  000277                    SCC
2236 007566  000244                    CLZ                     ;CC=1011
2237 007570  105724                    TSTB    (R4)+           ;
2238 007572  102403                    BVS     1$              ;BRANCH IF ERROR
2239 007574  103402                    BCS     1$
2240 007576  100401                    BMI     1$
2241 007600  001401                    BEQ     2$              ;BRANCH IF GOOD
2242 007602  104001            1$:     ERROR   +1              ;CPU ERROR
2243                                                           ;MODE 2 EVEN BYTE FAILED
2244 007604  000277            2$:     SCC
2245 007606  000250                    CLN                     ;CC=0111
2246 007610  105724                    TSTB    (R4)+           ;
2247 007612  001403                    BEQ     3$              ;
2248 007614  103402                    BCS     3$
2249 007616  102401                    BVS     3$
2250 007620  100401                    BMI     4$              ;BRANCH IF GOOD
2251 007622  104001            3$:     ERROR   +1              ;CPU ERROR
```

```
                                                                ;MODE 2 ODD BYTE FAILED
                        4$:
                        ;
                        MSPAA:
                        ;       TEST TST MODE 3
005004                          CLR     R4
005014                          CLR     (R4)
105114                          COMB    (R4)
005214                          INC     (R4)            ;0=400
005034                          CLR     @(R4)+          ;400=0
005004                          CLR     R4              ;R4=0
000277                          SCC
000244                          CLZ                     ;CC=1011
005734                          TST     @(R4)+          ; TEST MODE 3
103403                          BCS     1$              ;BRANCH IF ERROR
102402                          BVS     1$              ;
100401                          BMI     1$
001401                          BEQ     2$              ;BRANCH IF GOOD
104001                  1$:     ERROR   +1              ;CPU ERROR
                                                        ;MODE 3 FAILED
005304                  2$:     DEC     R4
005304                          DEC     R4              ;R4=0
005334                          DEC     @(R4)+          ;400=-1
005004                          CLR     R4
000277                          SCC
000250                          CLN                     ;CC=0111
005734                          TST     @(R4)+          ; TEST INSTRUCTION
103403                          BCS     3$              ;
001402                          BEQ     3$              ;BRANCH IF ERROR
102401                          BVS     3$              ;
100401                          BMI     4$              ;BRANCH IF GOOD
                                                        ;ERROR MODE 3
104001                  3$:     ERROR   +1              ;CPU ERROR
                        4$:
                        ;
                        MSPBB:
                        ;       TEST TST MODE 3 AUTO-INC
005004                          CLR     R4
005014                          CLR     (R4)            ;0=0
105114                          COMB    (R4)            ;
005214                          INC     (R4)            ;0=400
005001                          CLR     R1
105101                          COMB    R1
005201                          INC     R1              ;R1=400
005011                          CLR     (R1)            ;400=0
000277                          SCC
005734                          TST     @(R4)+          ;400=0
103403                          BCS     1$              ;ERROR IF CARRY
102402                          BVS     1$              ;ERROR IF OVERFLOW
100401                          BMI     1$              ;ERROR IF MINUS
001401                          BEQ     2$              ;ERROR IF NOT EQUAL
104001                  1$:     ERROR   +1              ;CPU ERROR
                                                ;CC SHOULD = 0100
```

M4

BASE INSTRUCTION SET TESTS

```
2313 007746  005304                2$:     DEC     R4
2314 007750  005304                        DEC     R4
2315 007752  005704                        TST     R4              ;SEE IF AUTO-INC WORKED
2316 007754  001401                        BEQ     4$              ;ERROR IF R4 NE 0
2317 007756  104001                3$:     ERROR   ;1              ;CPU ERROR
2318                                                        ;AUTO-INC FIALED
2319 007760                        4$:
2320
2321                               ;
2324 007760                        MSTB3:
2325
2326                               ;       TEST TST MODE 3 BYTE
2327 007760  005004                        CLR     R4
2328 007762  005014                        CLR     (R4)
2329 007764  105114                        COMB    (R4)
2330 007766  005214                        INC     (R4)
2331 007770  005214                        INC     (R4)            ;0=401
2332 007772  005001                        CLR     R1
2333 007774  105101                        COMB    R1
2334 007776  005201                        INC     R1              ;R1=400
2335 010000  005011                        CLR     (R1)
2336 010002  005111                        COM     (R1)
2337 010004  105011                        CLRB    (R1)            ;400=377 000
2338 010006  105734                        TSTB    @(R4)+          ;** TEST INSTRUCTION
2339 010010  001403                        BEQ     1$              ;ERROR IF EQUAL
2340 010012  103402                        BCS     1$              ;ERROR IF CARRY SET
2341 010014  102401                        BVS     1$              ;ERROR IR OVERFLOW
2342 010016  100401                        BMI     2$              ;BRANCH IF MINUS
2343 010020  104001                1$:     ERROR   ;1              ;CPU ERROR
2344                                                        ;CC ERROR
2345 010022  005304                2$:     DEC     R4
2346 010024  005304                        DEC     R4
2347 010026  001401                        BEQ     4$                      ;BRANCH IF AUTO-INC WORKED
2348 010030  104001                3$:     ERROR   ;1              ;CPU ERROR
2349                                                        ;AUTO-INC FAILED
2350 010032                        4$:
2351
2352                               ;
2355 010032                        MST4:
2356
2357                               ;       TEST TST MODE 4
2358 010032  005004                        CLR     R4
2359 010034  005014                        CLR     (R4)            ;0=0
2360 010036  005204                        INC     R4
2361 010040  005204                        INC     R4              ;R4=2
2362 010042  000277                        SCC
2363 010044  000244                        CLZ                     ;CC=1011
2364 010046  005744                        TST     -(R4)           ;**TEST INTRUCTION
2365 010050  103403                        BCS     1$              ;ERROR IF CARRY
2366 010052  102402                        BVS     1$              ;ERROR IF OVERFLOW
2367 010054  100401                        BMI     1$              ;ERROR IF MINUS
2368 010056  001401                        BEQ     2$              ;BRANCH IF GOOD
2369 010060  104001                1$:     ERROR   ;1              ;CPU ERROR
2370                                                        ;CC WRONG
2371 010062  005704                2$:     TST     R4              ;INSURE CORRECT AUTO-DEC
2372 010064  001401                        BEQ     4$              ;BRANCH IF GOOD AUTO-DEC
2373                                                               ;BAD AUTO-DEC
```

BASE INSTRUCTION SET TESTS

```
2374 010066  104001                3$:     ERROR   +1              ;CPU ERROR
2375 010070                        4$:
2376
2377                               ;
2380 010070                        MST4B:
2381
2382                               ;       TEST TST MODE 4 BYTE
2383 010070  005004                        CLR     R4
2384 010072  005014                        CLR     (R4)
2385 010074  005114                        COM     (R4)
2386 010076  105114                        COMB    (R4)            ;0=377 000
2387 010100  000277                        SCC
2388 010102  005204                        INC     R4
2389 010104  005204                        INC     R4              ;R4=2
2390 010106  105744                        TSTB    -(R4)           ;**TEST INSTRUCTION
2391 010110  001403                        BEQ     1$              ;ERROR IF EQUAL TO 0
2392 010112  103402                        BCS     1$              ;ERROR IF CARRY
2393 010114  102401                        BVS     1$              ;ERROR IF OVERFLOW
2394 010116  100401                        BMI     2$              ;BRANCH IF MINUS
2395 010120  104001                1$:     ERROR   +1              ;CPU ERROR
2396                                                       ;CC SHOULD EQUAL 0100
2397 010122  105744                2$:     TSTB    -(R4)           ;**TEST EVEN BYTE
2398 010124  001401                        BEQ     4$              ;BRANCH IF GOOD
2399 010126  104001                3$:     ERROR   +1              ;CPU ERROR
2400                                                       ;CC SHOULD EQUAL 0100 AND R4=-1
2401 010130                        4$:
2402                               ;
2405 010130                        MST5:
2406
2407                               ;       TEST TST MODE 5
2408 010130  005004                        CLR     R4              ;
2409 010132  005024                        CLR     (R4)+           ;0=0, R4=2
2410 010134  000277                        SCC
2411 010136  000244                        CLZ                     ;CC=1011
2412 010140  005754                        TST     @-(R4)          ; TEST INSTRUCTION
2413 010142  103403                        BCS     1$              ;ERROR IF CARRY
2414 010144  102402                        BVS     1$              ;ERROR IF OVERFLOW
2415 010146  100401                        BMI     1$              ;ERROR IF MINUS
2416 010150  001401                        BEQ     2$              ;BRANCH IF GOOD
2417 010152  104001                1$:     ERROR   +1              ;CPU ERROR
2418                                                       ;CC WRONG, SHOULD = 0100
2419 010154  005704                2$:     TST     R4
2420 010156  001401                        BEQ     4$              ;BRANCH IF AUTO-DEC WORKED
2421 010160  104001                3$:     ERROR   +1              ;CPU ERROR
2422                                                       ;AUTO-DEC FAILED
2423 010162                        4$:
2424
2425                               ;
2428 010162                        MST5B:
2429
2430                               ;       TEST TST MODE 5 BYTE
2431 010162  005004                        CLR     R4
2432 010164  005014                        CLR     (R4)
2433 010166  105114                        COMB    (R4)
2434 010170  005214                        INC     (R4)            ;0=400
2435 010172  005034                        CLR     @(R4)+          ;400=0, R4=2
2436 010174  005154                        COM     @-(R4)          ;
```

BASE INSTRUCTION SET TESTS

```
2437 010176  105134                        COMB    @(R4)+          ;400=377 000 R4=2
2438 010200  105754                        TSTB    @-(R4)          ;**TEST INSTRUCTION
2439 010202  103403                        BCS     1$              ;ERROR IF CARRY
2440 010204  100402                        BMI     1$              ;ERROR IF MINUS
2441 010206  102401                        BVS     1$              ;ERROR IF OVERFLOW
2442 010210  001401                        BEQ     2$              ;BRANCH IF GOOD
2443 010212  104001              1$:       ERROR   .1              ;CPU ERROR
2444                                                  ;CC SHOULD = 0100
2445 010214  005224              2$:       INC     (R4)+           ;0=401
2446 010216  105754                        TSTB    @-(R4)          ;**TEST INSTRUCTION
2447 010220  100401                        BMI     4$              ;BRANCH IF GOOD
2448                                                               ;EVEN BYTE FAILURE
2449 010222  104001              3$:       ERROR   .1              ;CPU ERROR
2450 010224                      4$:
2451
2452                             ;
2455 010224                      MST6:
2456
2457                             ;         TEST TST MODE 6
2458 010224  005004                        CLR     R4
2459 010226  005014                        CLR     (R4)            ;0=0
2460 010230  105104                        COMB    R4
2461 010232  005204                        INC     R4              ;R4=400
2462 010234  005014                        CLR     (R4)
2463 010236  005114                        COM     (R4)            ;400=-1
2464 010240  005764  177400                TST     -400(R4)        ;**TEST LOCATION 0
2465 010244  103403                        BCS     1$              ;ERROR IF CARRY
2466 010246  102402                        BVS     1$              ;ERROR IF OVERFLOW
2467 010250  100401                        BMI     1$              ;ERROR IF MINUS
2468 010252  001401                        BEQ     2$              ;BRANCH IF ZERO
2469 010254  104001              1$:       ERROR   .1              ;CPU ERROR
2470                                                  ;CC ARE WRONG
2471 010256  005004              2$:       CLR     R4              ;
2472 010260  005764  000400                TST     400(R4)         ;TST LOCATION 400
2473 010264  001401                        BEQ     3$              ;ERROR IF EQUAL
2474 010266  100401                        BMI     4$              ;BRANCH IF MINUS
2475 010270  104001              3$:       ERROR   .1              ;CPU ERROR
2476                                                  ;CC ERROR
2477 010272                      4$:
2478
2479                             ;
2482 010272                      MST7:
2483
2484                             ;         TEST TST MODE 7
2485 010272  005004                        CLR     R4
2486 010274  005014                        CLR     (R4)
2487 010276  005124                        COM     (R4)+           ;0=-1
2488 010300  005014                        CLR     (R4)            ;2=0
2489 010302  005002                        CLR     R2              ;R2=0
2490 010304  005004                        CLR     R4
2491 010306  105104                        COMB    R4
2492 010310  005204                        INC     R4              ;R4=400
2493 010312  005014                        CLR     (R4)            ;400=0
2494 010314  005774  177402                TST     @-376(R4)       ;**TEST LOCATION 0
2495 010320  103403                        BCS     1$              ;ERROR IF CARRY
2496 010322  102402                        BVS     1$              ;ERROR IF OVERFLOW
2497 010324  001401                        BEQ     1$              ;ERROR IF ZERO
```

```
                                        BMI     2$              ;BRANCH IF GOOD
                                1$:     ERROR   .1              ;CPU ERROR
                                                        ;ERROR IN CC
                                2$:     INC     (R2)+           ;0=-2 R2=2
                                        TST     @-376(R4)       ;** CHECK CONTENTS OF LOCATION 2
                                        BMI     3$              ;ERROR IF MINUS
                                        BEQ     4$              ;BRANCH IF GOOD
                                3$:     ERROR   .1              ;CPU ERROR
                                                        ;CC SHOULD = 0100
                                4$:
                                ;
                                ;
                                ;
                                ;
                                ;
                                .SBTTL ************ DOUBLE OPERAND TESTS ****************
                                ;
                                MDMO:
                                ;       TEST MOVE MODE 0
                                        CLR     R4              ;R4=0
                                        CLR     R1
                                        COM     R1              ;R1=-1
                                        MOV     R1,R4           ;**TEST MOVE INSTRUCTION
                                        BEQ     1$              ;ERROR IF R4=0
                                        BVS     1$              ;ERROR IF OBERFLOW
                                        BMI     2$              ;BRANCH IF MINUS
                                1$:     ERROR   .1              ;CPU ERROR
                                                        ;CC SHOULD = 100-, R4 SHOULD=-1
                                2$:     INC     R4              ;R4 SHOULD =0
                                        BNE     1$              ;ERROR IF R4 NE 0
                                ;
                                MDAO:
                                ;       TEST ADD MODE 0
                                        CLR     R4              ;R4=0
                                        CLR     R1
                                        COM     R1              ;R1=-1
                                        ADD     R1,R4           ;** TEST ADD OF R1 TO R4
                                        BEQ     1$              ;ERROR IF ZERO
                                        BCS     1$              ;ERROR IF CARRY
                                        BVS     1$              ;ERROR IF OVERFLOW
                                        BMI     2$              ;BRANCH IF MINUS
                                1$:     ERROR   .1              ;CPU ERROR
                                                        ;CC SHOULD = 1000 , R4=-1
                                2$:     INC     R4              ;R4 SHOULD =0
                                        BEQ     4$              ;BRANCH IF R4=0
                                3$:     ERROR   .1              ;CPU ERROR
                                                        ;R4 SHOULD = 0
                                4$:
                                ;
                                MDSO:
                                ;       TEST SUB MODE 0
```

```
********** DOUBLE OPERAND TESTS ****************

2561 010422  005004                        CLR     R4
2562 010424  005001                        CLR     R1
2563 010426  005201                        INC     R1                      ;R1=1 R4=0
2564 010430  160104                        SUB     R1,R4                   ;**TEST OF R4-R1, R4=-1
2565 010432  102403                        BVS     1$                      ;ERROR IF V SET
2566 010434  103002                        BCC     1$                      ;ERROR IF NO CARRY
2567 010436  001401                        BEQ     1$                      ;ERROR IF =0
2568 010440  100401                        BMI     2$                      ;BRANCH IF MINUS
2569 010442  104001                1$:     ERROR   ;1              ;CPU ERROR
2570                                                               ;CC SHOULD = 1001
2571 010444  005101                2$:     COM     R1                      ;R1=1
2572 010446  005201                        INC     R1                      ;GET TWO'S COMPLIMENT, R1=-1
2573 010450  160104                        SUB     R1,R4                   ;**TEST R4-R1 (1- 1= 0
2574 010452  001401                        BEQ     4$                      ;BRANCH IF ZERO
2575 010454  104001                3$:     ERROR   ;1              ;CPU ERROR
2576                                                               ;CC SHOULD = 0100
2577 010456                        4$:
2578
2579                                ;
2582 010456                        MDM27:
2583
2584                                ;       TEST MOV MODE 27,00
2585 010456  000257                        CCC                             ;CC=0000
2586 010460  012704  125252                MOV     #125252,R4              ;**TEST MOVE
2587 010464  001401                        BEQ     1$                      ;ERROR IF = 0
2588 010466  100401                        BMI     2$                      ;BRANCH IF MINUS
2589 010470  104001                1$:     ERROR   ;1              ;CPU ERROR
2590                                                               ;CC SHOULD = 1000
2591 010472  012701  052525        2$:     MOV     #052525,R1              ;**TEST MOVE
2592 010476  100401                        BMI     3$                      ;ERROR IF MINUS
2593 010500  001001                        BNE     4$                      ;BRANCH IF NE 0
2594 010502  104001                3$:     ERROR   ;1              ;CPU ERROR
2595                                                               ;CC SHOULD = 0000
2596 010504  060104                4$:     ADD     R1,R4                   ;R1+R4=-1
2597 010506  100401                        BMI     6$                      ;BRANCH IF MINUS
2598 010510  104001                5$:     ERROR   ;1              ;CPU ERROR
2599                                                               ;MOV FAILED
2600 010512  005204                6$:     INC     R4                      ;R4+1=0
2601 010514  001375                        BNE     5$                      ;ERROR IF NOT ZERO
2602
2603                                ;
2606 010516                        MBI00:
2607
2608                                ;       TEST BIC, BIS MODE 0,0
2609 010516  005004                        CLR     R4
2610 010520  005104                        COM     R4                      ;R4=-1
2611 010522  012701  125252                MOV     #125252,R1              ;SETUP R1 TEST DATA
2612 010526  012702  052525                MOV     #052525,R2              ;R2=COMPLIMENT OF R1
2613 010532  000261                        SEC
2614 010534  040104                        BIC     R1,R4                   ;**TEST BIC WITH CARRY SET
2615 010536  103003                        BCC     1$                      ;ERROR IF NO CARRY
2616 010540  102402                        BVS     1$                      ;ERROR IF OVERFLOW
2617 010542  001401                        BEQ     1$                      ;ERROR IF 0
2618 010544  100001                        BPL     2$                      ;BRANCH IF PLUS
2619 010546  104001                1$:     ERROR   ;1              ;CPU ERROR
2620                                                               ;CC SHOULD = 0001
2621 010550  020402                2$:     CMP     R4,R2                   ;COMPARE CONTENTS OF R4 AND R2
```

```
010552  001401                          BEQ     4$                      ;BRANCH IF EQUAL
010554  104001                  3$:     ERROR   +1              ;CPU ERROR
                                                                ;R4 AND R2 SHOULD BE EQUAL
010556  005301                  4$:     DEC     R1                      ;R1=125251
010560  050201                          BIS     R2,R1                   ;BIS 052525 AND 125251=177775
010562  100401                          BMI     6$                      ;BRANCH IF MIUS VALUE
010564  104001                  5$:     ERROR   +1              ;CPU ERROR
                                                                ;BAD BIS OPERATION
010566  005201                  6$:     INC     R1
010570  005201                          INC     R1
010572  005201                          INC     R1                      ;R1=0
010574  001373                          BNE     5$                      ;ERROR IF NE 0

                                ;
010576                          MBC00:

                                ;       TEST BIT, CMP MODE 0,0
010576  012701  125252                  MOV     #125252,R1              ;R1=125252
010602  012704  100000                  MOV     #100000,R4              ;R4=100000
010606  012702  052525                  MOV     #052525,R2              ;R2=052525
010612  030401                          BIT     R4,R1                   ;**TEST OF BIT ,CC=1000
010614  001401                          BEQ     1$                      ;ERROR IF EQ 0
010616  100401                          BMI     2$                      ;BRANCH IF GOOD
010620  104001                  1$:     ERROR   +1              ;CPU ERROR
                                                                ;CC SHOULD = 1000
010622  020401                  2$:     CMP     R4,R1                   ;*TEST 100000-125252=25252
010624  001402                          BEQ     3$                      ;ERROR IF EQUAL 0
010626  103001                          BCC     3$                      ;ERROR IF CARRY CLEARED
010630  100401                          BMI     4$                      ;BRANCH IF GOOD
010632  104001                  3$:     ERROR   +1              ;CPU ERROR
                                                                ;CC SHOULD = 0010
010634  020104                  4$:     CMP     R1,R4                   ;125252-100000 = 25252
010636  001403                          BEQ     5$                      ;ERROR IF EQUAL
010640  103402                          BCS     5$                      ;ERROR IF CARRY
010642  102401                          BVS     5$                      ;ERROR IF OVERFLOW
010644  100001                          BPL     6$                      ;BRANCH IF GOOD
010646  104001                  5$:     ERROR   +1              ;CPU ERROR
                                                                ;CC SHOULD =0001
010650  005004                  6$:     CLR     R4
010652  005204                          INC     R4                      ;R4=1
010654  000277                          SCC
010656  030401                          BIT     R4,R1                   ;R4 ↑ R1 = 2
010660  001401                          BEQ     8$                      ;BRANCH IF GOOD
010662  104001                  7$:     ERROR   +1              ;CPU ERROR
                                                                ;CC SHOULD = 0101
010664                          8$:

                                ;
010664                          MM11:

                                ;       TEST MOV, MOVB MODE 1,1 AND SIGN EXT ON MOVB TO GPR
010664  012704  000400                  MOV     #400,R4         ;R4=400
010670  012701  000402                  MOV     #402,R1         ;R1=402
010674  005014                          CLR     (R4)            ;
010676  005114                          COM     (R4)            ;400=-1
010700  005011                          CLR     (R1)                    ;
010702  105111                          COMB    (R1)            ;402=000 377
```

```
********** DOUBLE OPERAND TESTS ******************

 2683 010704  005002                       CLR     R2              ;R2=0
 2684 010706  012703  000405                MOV     #405,R3         ;R3=405
 2685 010712  000277                        SCC                     ;CC=1111
 2686 010714  011412                        MOV     (R4),(R2)       ;MOV    400 TO 0 ,0=-1
 2687 010716  001403                        BEQ     1$              ;ERROR IF 0
 2688 010720  102402                        BVS     1$              ;ERROR IF OVERFLOW
 2689 010722  103001                        BCC     1$              ;ERROR IF NO CARRY
 2690 010724  100401                        BMI     2$              ;BRANCH IF GOOD
 2691 010726  104001                1$:     ERROR   +1              ;CPU ERROR
 2692                                                    ;CC SHOULD =1001
 2693 010730  005212                2$:     INC     (R2)            ;0=0
 2694 010732  001004                        BNE     3$              ;ERROR IF NOT 0
 2695 010734  000257                        CCC                     ;CC=0000
 2696 010736  111113                        MOVB    (R1),(R3)               ;405=377
 2697 010740  001401                        BEQ     3$              ;ERROR IF EQUAL
 2698 010742  100401                        BMI     4$              ;BRANCH IF GOOD
 2699 010744  104001                3$:     ERROR   +1              ;CPU ERROR
 2700                                                    ;
 2701 010746  105213                4$:     INCB    (R3)            ;405=0
 2702 010750  001375                        BNE     3$              ;ERROR IF 405 NOT 0
 2703                                ;CHECK THAT SIGN EXTENSION OCCURS ON A MOVB TO GENERAL REGISTER.
 2704 010752  005002                        CLR     R2              ;INIT R2 TO ZERO.
 2705 010754  111102                        MOVB    (R1),R2         ;MOVE 377 TO R2
 2706 010756  100005                        BPL     5$              ;ERROR! BIT 15 SHOULD BE SET.
 2707 010760  102404                        BVS     5$              ;V BIT SHOULD BE CLEARED
 2708 010762  103403                        BCS     5$              ;CARRY BIT SHOULD BE UNAFFECTED
 2709 010764  022702  177777                CMP     #177777,R2      ;TEST R2
 2710 010770  001401                        BEQ     6$              ;SIGN EXTENDED THROUGH UPPER BYTE
 2711                                                                ;ERROR! BYTE SHOULD HAVE
 2712                                                                ;SIGN EXTENDED THROUGH UPPER BYTE
 2713 010772  104001                5$:     ERROR   +1              ;CPU ERROR
 2714 010774                        6$:
 2715
 2716                                ;
 2719 010774                        MA11:
 2720
 2721                                ;       TEST ADD MODE 1,1
 2722 010774  012704  000400                MOV     #400,R4         ;R4=400
 2723 011000  012701  000402                MOV     #402,R1         ;R1=402
 2724 011004  012714  177753                MOV     #-25,(R4)       ;400=-25
 2725 011010  012711  000024                MOV     #24,(R1)        ;402=24
 2726 011014  061114                        ADD     (R1),(R4)       ;-25+24=-1
 2727 011016  001404                        BEQ     1$              ;ERROR IF 0
 2728 011020  103403                        BCS     1$              ;ERROR IF CARRY
 2729 011022  100002                        BPL     1$              ;ERROR IF POSITIVE RESULT
 2730 011024  005214                        INC     (R4)            ;-1+1=0
 2731 011026  001401                        BEQ     2$              ;BRANCH IF GOOD
 2732 011030  104001                1$:     ERROR   +1              ;CPU ERROR
 2733                                                    ;CC SHOULD = 1000
 2734 011032                        2$:
 2735
 2736                                ;
 2739 011032                        MS11:
 2740
 2741                                ;       TEST SUB MODE 1,1
 2742 011032  012704  000400                MOV     #400,R4         ;R4=400
 2743 011036  012701  000404                MOV     #404,R1         ;R1=404
```

```
********** DOUBLE OPERAND TESTS ******************

 2744 011042  012714  000003                MOV     #3,(R4)          ;400=3
 2745 011046  012711  000006                MOV     #6,(R1)          ;406=6
 2746 011052  000277                        SCC                      ;CC=1111
 2747 011054  161411                        SUB     (R4),(R1)        ;6-3=3
 2748 011056  001402                        BEQ     1$               ;ERROR IF 0
 2749 011060  100401                        BMI     1$               ;ERROR IF MINUS
 2750 011062  103001                        BCC     2$               ;BRANCH IF GOOD
 2751 011064  104001              1$:       ERROR   +1               ;CPU ERROR
 2752                                                  ;CC SHOULD = 0000
 2753 011066  161411              2$:       SUB     (R4),(R1)        ;3-3=0
 2754 011070  001375                        BNE     1$               ;ERROR IF NOT 0
 2755
 2756                              ;
 2759 011072                      MBB11:
 2760
 2761                              ;       TEST BIC, BIS MODE 1,1
 2762 011072  012704  000400                MOV     #400,R4          ;R4=400
 2763 011076  012701  000402                MOV     #402,R1          ;R1=402
 2764 011102  012714  052525                MOV     #052525,(R4)     ;400=052525
 2765 011106  012711  125252                MOV     #125252,(R1)     ;402=125252
 2766 011112  051411                        BIS     (R4),(R1)        ;R4 V R1 = -1
 2767 011114  001401                        BEQ     1$               ;ERROR IF 0
 2768 011116  100401                        BMI     2$               ;BRANCH IF GOOD
 2769 011120  104001              1$:       ERROR   +1               ;CPU ERROR
 2770                                                  ;CC SHOULD = 1000
 2771 011122  005211              2$:       INC     (R1)             ;402=0
 2772 011124  001401                        BEQ     4$               ;BRANCH IF GOOD
 2773 011126  104001              3$:       ERROR   +1               ;CPU ERROR
 2774                                                  ;CC SHOULD = 0100
 2775 011130  005311              4$:       DEC     (R1)             ;402=-1
 2776 011132  041411                        BIC     (R4),(R1)        ;R1=125252
 2777 011134  001401                        BEQ     5$               ;ERROR IF 0
 2778 011136  100401                        BMI     6$               ;BRANCH IF GOOD
 2779 011140  104001              5$:       ERROR   +1               ;CPU ERROR
 2780                                                  ;CC SHOULD = 1000
 2781 011142  005111              6$:       COM     (R1)             ;402=052525
 2782 011144  041114                        BIC     (R1),(R4)        ;400=0
 2783 011146  001401                        BEQ     8$               ;BRANCH IF GOOD
 2784 011150  104001              7$:       ERROR   +1               ;CPU ERROR
 2785                                                  ;CC SHOULD = 0100
 2786 011152                      8$:
 2787
 2788                              ;
 2791 011152                      MBC11:
 2792
 2793                              ;       TEST BIT, CMP MODE 1,1
 2794 011152  012704  000400                MOV     #400,R4                  ;R4=400
 2795 011156  012714  052525                MOV     #052525,(R4)             ;400=052525
 2796 011162  012701  000402                MOV     #402,R1                  ;R1-402
 2797 011166  012711  125252                MOV     #125252,(R1)             ;402=125252
 2798 011172  000241                        CLC                              ;CLEAR CARRY
 2799 011174  031411                        BIT     (R4),(R1)                ;**052525↑125252=0
 2800 011176  103401                        BCS     1$                       ;ERROR IF CARRY
 2801 011200  001401                        BEQ     2$                       ;BRANCH IF GOOD
 2802 011202  104001              1$:       ERROR   +1               ;CPU ERROR
 2803                                                  ;CC SHOULD = 0100
 2804 011204  021411              2$:       CMP     (R4),(R1)                ;400-402=125253
```

```
********** DOUBLE OPERAND TESTS ******************

 2805 011206  001403                        BEQ     3$                      ;ERROR IF ZERO
 2806 011210  103002                        BCC     3$                      ;ERROR IF NO CARRY
 2807 011212  102001                        BVC     3$                      ;ERROR IF NO OVERFLOW
 2808 011214  100401                        BMI     4$                      ;BRANCH IF GOOD
 2809 011216  104001              3$:       ERROR   +1              ;CPU ERROR
 2810                                                               ;CC SHOULD = 1000
 2811 011220  005014              4$:       CLR     (R4)
 2812 011222  005214                        INC     (R4)                    ;400=1
 2813 011224  031114                        BIT     (R1),(R4)               ;125252↑1=0
 2814 011226  001401                        BEQ     6$                      ;BRANCH IF GOOD
 2815 011230  104001              5$:       ERROR   +1              ;CPU ERROR
 2816                                                               ;CC SHOULD= 0100
 2817 011232                      6$:
 2818
 2819                             ;
 2822 011232                      MM22:
 2823
 2824                             ;       TEST MOV MODE 2,2
 2825 011232  012704  000400                MOV     #400,R4         ;R4=400
 2826 011236  012701  000402                MOV     #402,R1         ;R1=402
 2827 011242  012714  000005                MOV     #5,(R4)         ;400=5
 2828 011246  005021                        CLR     (R1)+           ;402=0
 2829 011250  005011                        CLR     (R1)            ;
 2830 011252  005111                        COM     (R1)            ;404=-1
 2831 011254  005741                        TST     -(R1)           ;R1=402
 2832 011256  000277                        SCC                     ;CC=1111
 2833 011260  012124                        MOV     (R1)+,(R4)+     ;400=0 R4=402 R1=404
 2834 011262  100403                        BMI     1$              ;ERROR IF MINUS
 2835 011264  103002                        BCC     1$              ;ERROR IF NO CARRY
 2836 011266  102401                        BVS     1$              ;ERROR IF OVERFLOW
 2837 011270  001401                        BEQ     2$              ;BRANCH IF GOOD
 2838 011272  104001              1$:       ERROR   +1              ;CPU ERROR
 2839                                                       ;CC SHOULD= 0101
 2840 011274  005244              2$:       INC     -(R4)           ;400=1 R4=400
 2841
 2842 011276  061411                        ADD     (R4),(R1)       ;1+ -1 =0
 2843 011300  001401                        BEQ     4$              ;BRANCH IF GOOD
 2844 011302  104001              3$:       ERROR   +1              ;CPU ERROR
 2845                                                       ;CC SHOULD = 0100
 2846 011304                      4$:
 2847
 2848                             ;
 2851 011304                      MS22:
 2852
 2853                             ;       TEST SUB MODE 2,2
 2854 011304  012704  000400                MOV     #400,R4                 ;R4=400
 2855 011310  012701  000402                MOV     #402,R1                 ;R1=402
 2856 011314  012714  177760                MOV     #177760,(R4)            ;400=177760
 2857 011320  012711  177750                MOV     #177750,(R1)            ;402=177750
 2858 011324  162421                        SUB     (R4)+,(R1)+             ;R1=177770
 2859 011326  001403                        BEQ     1$                      ;ERROR IF ZERO
 2860 011330  102402                        BVS     1$                      ;ERROR IF OVERFLOW
 2861 011332  103001                        BCC     1$                      ;ERROR IF NO CARRY
 2862 011334  100401                        BMI     2$                      ;BRANCH IF GOOD
 2863 011336  104001              1$:       ERROR   +1              ;CPU ERROR
 2864                                                               ;CC SHOULD=1000
 2865 011340  005241              2$:       INC     -(R1)                   ;R1=1777771
```

```
********** DOUBLE OPERAND TESTS ******************

 2866 011342  162721  177771                SUB     #177771,(R1)+           ;R1=0
 2867 011346  100401                        BMI     3$                      ;ERROR IF MINUS
 2868 011350  001401                        BEQ     4$                      ;BRANCH IF GOOD
 2869 011352  104001                3$:     ERROR   +1              ;CPU ERROR
 2870                                                               ;CCSHOULD = 0100
 2871 011354                        4$:
 2872
 2873                               ;
 2889 011354                        MBB22:
 2890
 2891                               ;       TEST BIC, BICB, BIS, BISB MODE 2,2
 2892 011354  012704  000400                MOV     #400,R4                 ;R4=400
 2893 011360  012701  000402                MOV     #402,R1                 ;R1=402
 2894 011364  012702  000404                MOV     #404,R2                 ;R2=404
 2895 011370  012714  141401                MOV     #141401,(R4)            ;400=303 001
 2896 011374  012711  177405                MOV     #177405,(R1)            ;402=377 005
 2897 011400  012722  000070                MOV     #70,(R2)+               ;404=2070
 2898 011404  012722  177777                MOV     #-1,(R2)+               ;406=-1
 2899 011410  042421                        BIC     (R4)+,(R1)+             ;402=074004
 2900 011412  001401                        BEQ     1$                      ;ERROR IF ZERO
 2901 011414  100001                        BPL     2$                      ;BRANCH IF GOOD
 2902                                                               ;CC SHOULD = 1000
 2903 011416  104001                1$:     ERROR   +1                      ;CPU ERROR
 2904 011420  052421                2$:     BIS     (R4)+,(R1)+             ;404=074074
 2905 011422  142421                        BICB    (R4)+,(R1)+             ;406=074
 2906 011424  005301                        DEC     R1                      ;R4=405 R1=406
 2907 011426  152421                        BISB    (R4)+,(R1)+             ;406=-1 R4=406 R1=407
 2908 011430  100401                        BMI     4$                      ;BRANCH IF GOOD
 2909                                                               ;406 SHOULD=-1
 2910 011432  104001                3$:     ERROR   +1                      ;CPU ERROR
 2911 011434  005214                4$:     INC     (R4)                    ;406 SHOULD=0
 2912 011436  001401                        BEQ     6$                      ;BRANCH IF GOOD
 2913                                                               ;ERROR! 406 NE 0
 2914 011440  104001                5$:     ERROR   +1                      ;CPU ERROR
 2915 011442                        6$:
 2916
 2917                               ;
 2920 011442                        MBC22:
 2921
 2922                               ;       TEST BIT, CMP MODE 2,2
 2923 011442  012704  000400                MOV     #400,R4                 ;R4=400
 2924 011446  012701  000402                MOV     #402,R1                 ;R1=402
 2925 011452  012714  125252                MOV     #125252,(R4)            ;400=125252
 2926 011456  012721  100001                MOV     #100001,(R1)+           ;402=100001
 2927 011462  012711  100002                MOV     #100002,(R1)            ;404=100002
 2928 011466  005741                        TST     -(R1)                   ;R1=402
 2929 011470  132421                        BITB    (R4)+,(R1)+             ;**ANDED RESULT= 000
 2930 011472  100401                        BMI     1$                      ;ERROR IF MINUS
 2931 011474  001401                        BEQ     2$                      ;BRANCH IF GOOD
 2932 011476  104001                1$:     ERROR   +1                      ;CPU ERROR
 2933                                                       ;CC SHOULD = 0100
 2934 011500  132124                2$:     BITB    (R1)+,(R4)+             ;** ANDED RESULT = 200
 2935 011502  001401                        BEQ     3$                      ;ERROR IF EQUAL
 2936 011504  100401                        BMI     4$                      ;BRANCH IF GOOD
 2937 011506  104001                3$:     ERROR   +1                      ;CPU ERROR
 2938                                                       ;CC SHOULD= 1000 R4=402 R1=404
 2939 011510  022421                4$:     CMP     (R4)+,(R1)+             ;RESULT =+1
```

```
********** DOUBLE OPERAND TESTS ******************

 2940 011512  001402                        BEQ     5$              ;ERROR IF EQUAL
 2941 011514  103001                        BCC     5$              ;ERROR IF NO CARRY
 2942 011516  100401                        BMI     6$              ;BRANCH IF GOOD
 2943 011520  104001                5$:     ERROR   +1              ;CPU ERROR
 2944                                                       ;CC SHOULD = 0000
 2945 011522  005341                6$:     DEC     -(R1)           ;404=100001
 2946 011524  005741                        TST     -(R1)           ;R4=404 R1=402
 2947 011526  022124                        CMP     (R1)+,(R4)+     ;RESULT =0
 2948 011530  001401                        BEQ     8$              ;BRANCH IF GOOD
 2949                                                               ;CC SHOULD = 0100 R1=404 R4=406
 2950 011532  104001                7$:     ERROR   +1              ;CPU ERROR
 2951 011534                        8$:
 2952
 2953                                ;
 2956 011534                        MS33:
 2957
 2958                                ;       TEST SUB MODE 3,3
 2959 011534  005004                        CLR     R4              ;R4=0
 2960 011536  012701  000002                MOV     #2,R1           ;R1=2
 2961 011542  012702  000400                MOV     #400,R2         ;R2=400
 2962 011546  012714  000400                MOV     #400,(R4)       ;0=400
 2963 011552  012711  000402                MOV     #402,(R1)       ;2=402
 2964 011556  012722  000200                MOV     #200,(R2)+      ;400=200
 2965 011562  012712  054320                MOV     #54320,(R2)     ;402=54320
 2966 011566  163431                        SUB     @(R4)+,@(R1)+   ;54320 - 200=54120
 2967 011570  001402                        BEQ     1$              ;ERROR IF ZERO
 2968 011572  103401                        BCS     1$              ;ERROR IF CARRY
 2969 011574  100001                        BPL     2$              ;BRANCH IF GOOD
 2970 011576  104001                1$:     ERROR   +1              ;CPU ERROR
 2971                                                       ;CC SHOULD =0001
 2972 011600  022712  054120        2$:     CMP     #54120,(R2)     ; TEST R4 AUTO-INC AND RESULT
 2973 011604  001401                        BEQ     4$              ;BRANCH IF GOOD
 2974 011606  104001                3$:     ERROR   +1              ;CPU ERROR
 2975                                                       ;CC SHOULD = 0100 R4=2 R1=4
 2976 011610  005067  166164        4$:     CLR     0               ;RESTORE VECTORS
 2977 011614  005067  166162                CLR     2               ;       "
 2978
 2979
 2980                                ;
 2983 011620                        MCB44:
 2984
 2985                                ;       TEST CMP, BIT MODE 4,4
 2986 011620  012704  000400                MOV     #400,R4                 ;R4=400
 2987 011624  012701  000402                MOV     #402,R1                 ;R1=402
 2988 011630  012721  125366                MOV     #125366,(R1)+           ;402=125366 R1=404
 2989 011634  012724  173001                MOV     #173001,(R4)+           ;400=173001 R4=402
 2990 011640  024441                        CMP     -(R4),-(R1)             ;173001 - 125366=045603 CC=0000
 2991 011642  103401                        BCS     1$                      ;ERROR IF CARRY
 2992 011644  100001                        BPL     2$                      ;BRANCH IF GOOD
 2993                                                                       ;BAD COMPARE
 2994 011646  104001                1$:     ERROR   +1              ;CPU ERROR
 2995 011650  005204                2$:     INC     R4
 2996 011652  005201                        INC     R1                      ;R1=403
 2997 011654  000261                        SEC                             ;SET CARRY
 2998 011656  134144                        BITB    -(R1),-(R4)             ;173↑1=0
 2999 011660  103001                        BCC     3$                      ;C SHOULD REMAIN SET
 3000 011662  001401                        BEQ     4$                      ;BRANCH IF GOOD
```

```
********** DOUBLE OPERAND TESTS ******************

3001                                                                    ;INCORRECT COMPARE R4=400 R1=402
3002 011664  104001                    3$:     ERROR   .1              ;CPU ERROR
3003 011666  005724                    4$:     TST     (R4).           ;R4=402
3004 011670  005201                            INC     R1              ;R1=403
3005 011672  124441                            CMPB    -(R4),-(R1)     ;173 - 173=0
3006 011674  001401                            BEQ     6$              ;BRANCH IF GOOD
3007                                                                    ;BAD COMPARE
3008 011676  104001                    5$:     ERROR   .1              ;CPU ERROR
3009 011700                            6$:
3010                                   ;
3013 011700                            MA55:
3014
3015                                   ;       TEST ADD MODE 5,5
3016 011700  012704  000400                     MOV     #400,R4
3017 011704  012724  000001                     MOV     #1,(R4).        ;400=1
3018 011710  012724  177776                     MOV     #-2,(R4).       ;402=-2
3019 011714  012724  000400                     MOV     #400,(R4).      ;404=400
3020 011720  012714  000402                     MOV     #402,(R4)       ;406=402 R4=406
3021 011724  012701  000410                     MOV     #410,R1         ;R1=410
3022 011730  065451                             ADD     @-(R4),@-(R1)   ;1. -2= -1
3023 011732  001402                             BEQ     1$              ;ERROR IF ZERO
3024 011734  100001                             BPL     1$              ;ERROR IF PLUS
3025 011736  103001                             BCC     2$              ;BRANCH IF GOOD
3026 011740  104001                    1$:     ERROR   .1              ;CPU ERROR
3027                                                                    ;BAD ADD
3028 011742  062704  000004            2$:     ADD     #4,R4           ;R4=410
3029 011746  065154                             ADD     @-(R1),@-(R4)   ;-1 . 1 = 0
3030 011750  001401                             BEQ     4$              ;BRANCH IF GOOD
3031 011752  104001                    3$:     ERROR   .1              ;CPU ERROR
3032                                                                    ;CC SHOULD= 0100 R4=406 R1=402
3033 011754                            4$:
3034
3035                                   ;
3038                                   ;
3039                                   ;
3040                                   ;-----------------------------------
3041                                   ;TEST DOP BIT(B) MODE 6,6
3042                                   ;
3043                                           ;
3044                                           ;
3045                                           ;
3046 011754                            MB66:
3047
3048                                   ;       TEST BIT, BITB MODE 6,6
3049 011754  005004                             CLR     R4              ;R4=0
3050 011756  012701  000400                     MOV     #400,R1         ;
3051 011762  012721  125252                     MOV     #125252,(R1).   ;400=125252
3052 011766  012721  000001                     MOV     #1,(R1).        ;402=1
3053 011772  012721  100000                     MOV     #100000,(R1).   ;404=1000000  R1=406
3054 011776  036461  000400  177774            BIT     400(R4),-4(R1)  ;(400)&(402)=0
3055 012004  001401                             BEQ     2$              ;BRANCH IF GOOD
3056                                                                    ;CC SHOULD = 0100
3057 012006  104001                    1$:     ERROR   .1              ;CPU ERROR
3058 012010  136461  000405  177772    2$:     BITB    405(R4),-6(R1)  ;(405)&(400)=200
3059 012016  001401                             BEQ     3$              ;ERROR IF ZERO
3060 012020  100401                             BMI     4$              ;BRANCH IF GOOD
3061                                                                    ;CC SHOULD = 1000
```

```
********** DOUBLE OPERAND TESTS ******************

3062 012022  104001                  3$:     ERROR   ;1              ;CPU ERROR
3063 012024                          4$:
3064
3065                                 ;
3068 012024                          MS77:
3069
3070                                 ;       TEST SUB MODE 7,7
3071 012024  012704  000400                     MOV     #400,R4         ;
3072 012030  005001                             CLR     R1
3073 012032  012724  177776                     MOV     #-2,(R4)+               ;400=-2
3074 012036  012724  177777                     MOV     #-1,(R4)+               ;402=-1
3075 012042  012724  000400                     MOV     #400,(R4)+              ;404=400 R4=406
3076 012046  012711  000402                     MOV     #402,(R1)               ;0=402
3077 012052  005201                             INC     R1                      ;R1=1
3078 012054  167471  177372  000403             SUB     @-406(R4),@403(R1)      ;-2 - -1 = -1
3079 012062  001401                             BEQ     1$                      ;ERROR  IF ZERO
3080 012064  100401                             BMI     2$                      ;BRANCH IF GOOD
3081                                                                             ;CC SHOULD=1000
3082 012066  104001                  1$:     ERROR   ;1              ;CPU ERROR
3083 012070  167174  177777  177776  2$:     SUB     @-1(R1),@-2(R4)         ;-1 - -1 = 0
3084 012076  001401                             BEQ     4$                      ;BRANCH IF GOOD
3085 012100  104001                  3$:     ERROR   ;1              ;CPU ERROR
3086                                                                    ;ERROR ON SUBTRACT 400-0
3087 012102  005067  165672          4$:     CLR     0                       ;RESTORE VECTORS
3088 012106  005067  165670                  CLR     2                       ;       "
3089
3090
3091                                 ;
3094 012112                          MRL0:
3095
3096                                 ;       TEST ROL, ROLB MODE 0
3097 012112  012704  125252                     MOV     #125252,R4              ;R4=125252
3098 012116  000277                             SCC                             ;CC=1111
3099 012120  006104                             ROL     R4                      ;R4=052525 WITH CARRY SET
3100 012122  102004                             BVC     1$                      ;ERROR IF V CLEAR
3101 012124  103003                             BCC     1$                      ;ERROR IF CARRY CLEAR
3102 012126  022704  052525                     CMP     #052525,R4              ;SEE IF R0 = EXPECTED
3103 012132  001401                             BEQ     2$                      ;ERROR IF R4 NE EXPECTD
3104 012134  104001                  1$:     ERROR   ;1              ;CPU ERROR
3105                                                                    ;ROL FAILED, CC SHOULD=0011
3106 012136  012704  125252          2$:     MOV     #125252,R4              ;R4=125252
3107 012142  000257                             CCC                             ;CC=0000
3108 012144  106104                             ROLB    R4                      ;ROTATE EVEN BYTE
3109 012146  103005                             BCC     3$                      ;ERROR IF NO CARRY
3110 012150  102004                             BVC     3$                      ;ERROR IF NO OVERFLOW
3111 012152  100403                             BMI     3$                      ;ERROR IF MINUS
3112 012154  022704  125124                     CMP     #125124,R4              ;SEE IF R4 = EXPECTED
3113 012160  001401                             BEQ     4$                      ;BRANCH IF GOOD
3114 012162  104001                  3$:     ERROR   ;1              ;CPU ERROR
3115                                                                    ;ROLB FAILED, CC SHOULD=1011, R4=125125
3116 012164                          4$:
3117
3118                                 ;
3121 012164                          MRLB1:
3122
3123                                 ;       TEST ROL, ROLB MODE 1
3124 012164  005004                             CLR     R4                      ;R4=0
```

```
012166  012714  052525                  MOV     #52525,(R4)             ;0=52525
012172  006114                          ROL     (R4)                    ;**TEST INSTRUCTION, 0=125252
012174  100005                          BPL     1$                      ;ERROR IF PLUS
012176  102004                          BVC     1$                      ;ERROR IF NO OVERFLOW
012200  103403                          BCS     1$                      ;ERROR IF CARRY
012202  021427  125252                  CMP     (R4),#125252            ;SEE IF R4=EXPECTED
012206  001401                          BEQ     2$                      ;BRANCH IF GOOD
                                                                        ;BAD ROL ,CC SHOULD=1010
012210  104001                  1$:     ERROR   +1                ;CPU ERROR
012212  012714  125252          2$:     MOV     #125252,(R4)            ;0=125252
012216  005204                          INC     R4                      ;R4=1
012220  000277                          SCC                             ;CC=1111
012222  106114                          ROLB    (R4)                    ;**TEST INSTRUCTION
012224  100406                          BMI     3$                      ;ERROR IF RESULT IS POSITIVE
012226  103005                          BCC     3$                      ;ERROR IF NO CARRY
012230  102004                          BVC     3$                      ;ERROR IF V CLEAR
012232  005304                          DEC     R4                      ;R4=0
012234  022714  052652                  CMP     #52652,(R4)             ;ERROR IF 0 NE EXPECTED
012240  001401                          BEQ     4$                      ;BRANCH IF GOOD
012242  104001                  3$:     ERROR   +1                ;CPU ERROR
                                                                  ;BAD ROLB ODD BYTE,CC SHOULD=1011
012244                          4$:

                                ;
012244                          MRL2:

                                ;       TEST ROL, ROLB MODE 2
012244  005004                          CLR     R4                      ;R4=0
012246  012714  100000                  MOV     #100000,(R4)            ;0=100000
012252  000257                          CCC                             ;CC=0000
012254  006124                          ROL     (R4)+                   ;*TEST INSTRUCTION
012256  103002                          BCC     1$                      ;ERROR IF NO CARRY
012260  102001                          BVC     1$                      ;ERROR IF NO OVERFLOW
012262  001401                          BEQ     2$                      ;BRANCH IF GOOD
                                                                        ;ROL FAILED ,CCSHOULD= 0100
012264  104001                  1$:     ERROR   +1                ;CPU ERROR
012266  005304                  2$:     DEC     R4
012270  005304                          DEC     R4
012272  001012                          BNE     3$                      ;ERROR IN AUTO-DEC
012274  012714  004040                  MOV     #4040,(R4)              ;0=4040
012300  000241                          CLC
012302  106124                          ROLB    (R4)+                   ;**TEST INSTURCTION
012304  103405                          BCS     3$                      ;ERROR IF CARRY SET
012306  102404                          BVS     3$                      ;ERROR IF V
012310  005304                          DEC     R4
012312  022714  004100                  CMP     #04100,(R4)             ;SEE IF 0= EXPECTED RESULT
012316  001401                          BEQ     4$                      ;BRANCH IF GOOD
012320  104001                  3$:     ERROR   +1                ;CPU ERROR
                                                                  ;BAD ROL
012322                          4$:

                                ;
012322                          MRL3:

                                ;       TEST ROL, ROLB MODE 3
012322  005004                          CLR     R4                      ;R4=0
012324  012714  052525                  MOV     #052525,(R4)                    ;0=52525
```

```
012330  000277                          SCC                             ;CC=1111
012332  006137  000000                  ROL     @#0                     ;**TEST INSRUCTION MODE 3 WITH PC
012336  100005                          BPL     1$                      ;ERROR IF PLUS
012340  102004                          BVC     1$                      ;ERROR IF NO OVERFLOW
012342  103403                          BCS     1$                      ;ERROR IF CARRY
012344  022714  125253                  CMP     #125253,(R4)            ;COMPARE RESULT WITH EXPECTED
012350  001401                          BEQ     2$                      ;BRANCH IF GOOD
                                                                        ;BAD ROL CC SHOULD=1010
012352  104001                  1$:     ERROR   +1                  ;CPU ERROR
012354  012714  125252          2$:     MOV     #125252,(R4)            ;0=125252
012360  000261                          SEC                             ;CC=---1
012362  106137  000000                  ROLB    @#0                     ;**TEST INSTRUCTION
012366  100402                          BMI     3$                      ;ERROR IF MINUS
012370  103001                          BCC     3$                      ;ERROR IF NO CARRY
012372  102401                          BVS     4$                      ;BRANCH IF OVERFLOW
012374  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ROL, CC SHOULD=1011
012376                          4$:

                                ;
012376                          MRL4:

                                ;       TEST ROL MODE 4
012376  005001                          CLR     R1                  ;R1=0
012400  012704  000002                  MOV     #2,R4               ;R4=2
012404  012711  054321                  MOV     #54321,(R1)         ;0=54321
012410  000277                          SCC                         ;CC=1111
012412  006144                          ROL     -(R4)               ;**TEST INSTRUCTION
012414  100007                          BPL     1$                  ;ERROR IF PLUS
012416  102006                          BVC     1$                  ;ERROR IF NO OVERFLOW
012420  103405                          BCS     1$                  ;ERROR IF CARRY
012422  022711  130643                  CMP     #130643,(R1)        ;SEE IF EXPECTED RESULT
012426  001002                          BNE     1$                  ;BRANCH IF ROL FAILED
012430  005704                          TST     R4                  ;SEE IF AUTO-DEC WORKED
012432  001401                          BEQ     2$                  ;BRANCH IF GOOD
                                                                    ;ERROR! BAD ROL INST
012434  104001                  1$:     ERROR   +1                  ;CPU ERROR
012436                          2$:

                                ;
012436                          MMRL5:

                                ;       TEST ROL MODE 5
012436  005004                          CLR     R4                      ;R4=0
012440  012714  000400                  MOV     #400,(R4)               ;0=400
012444  012734  123456                  MOV     #123456,@(R4)+          ;400=123465, R4=2
012450  000277                          SCC                             ;CC=1111
012452  006154                          ROL     @-(R4)                  ;**TEST INSTRUCTION
012454  100410                          BMI     1$                      ;ERROR IF RESULT IS MINUS
012456  103007                          BCC     1$                      ;ERROR IF NO CARRY
012460  102006                          BVC     1$                      ;ERROR IF NO OVERFLOW
012462  005704                          TST     R4                      ;SEE IF AUTO-DEC WORKED
012464  001004                          BNE     1$                      ;ERROR OF AUTO-DEC
012466  022737  047135  000400          CMP     #47135,@#400            ;SEE  IF CORRECT RESULT
012474  001401                          BEQ     2$                      ;BRANCH IF GOOD
                                                                        ;BAD ROL MODE 5
012476  104001                  1$:     ERROR   +1                  ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS ******************

3247 012500                          2$:
3248
3249                                 ;
3252 012500                          MRL6:
3253
3254                                 ;       TEST ROL MODE 6
3255 012500  012704  000400                  MOV     #400,R4                 ;R4=400
3256 012504  005001                          CLR     R1                      ;R1=0
3257 012506  012711  032525                  MOV     #32525,(R1)             ;0=32525
3258 012512  000277                          SCC
3259 012514  006164  177400                  ROL     -400(R4)                ;**TEST INSTRUCTION
3260 012520  100405                          BMI     1$                      ;ERROR IF MINUS
3261 012522  103404                          BCS     1$                      ;ERROR IF CARRY
3262 012524  102403                          BVS     1$                      ;ERROR IF OVERFLOW
3263 012526  022711  065253                  CMP     #65253,(R1)             ;SEE IF CORRECT RESULT
3264 012532  001401                          BEQ     2$                      ;BRANCH IF GOOD
3265                                                                         ;BAD ROL MODE 6
3266 012534  104001                  1$:     ERROR   .1              ;CPU ERROR
3267 012536                          2$:
3268
3269                                 ;
3272 012536                          MRL7:
3273
3274                                 ;       TEST ROL MODE 7
3275 012536  012704  000400                  MOV     #400,R4                 ;R4=400
3276 012542  005037  000402                  CLR     @#402                   ;402=0
3277 012546  012737  100000  000000          MOV     #100000,@#0             ;0=100000
3278 012554  006174  000002                  ROL     2(R4)                   ;**TEST INSTRUCTION
3279 012560  100406                          BMI     1$                      ;ERROR IF MINUS
3280 012562  001005                          BNE     1$                      ;ERROR IF NOT ZERO
3281 012564  103004                          BCC     1$                      ;ERROR IF NO CARRY
3282 012566  102003                          BVC     1$                      ;ERROR IF NO OVERFLOW
3283 012570  005737  000000                  TST     @#0                     ;CHECK RESULT
3284 012574  001401                          BEQ     2$                      ;BRANCH IF GOOD
3285                                                                         ;BAD ROL MODE 7
3286 012576  104001                  1$:     ERROR   .1              ;CPU ERROR
3287 012600                          2$:
3288
3289                                 ;
3350 012600                          MSW37:
3351
3352                                 ;       TEST SWAB MODE 37
3353 012600  012704  000400                  MOV     #400,R4                 ;R4=400
3354 012604  012714  040700                  MOV     #40700,(R4)             ;400= 101 300
3355 012610  000337  000400                  SWAB    @#400                   ;400 SHOULD = 300 101
3356 012614  100406                          BMI     1$                      ;ERROR IF MINUS
3357 012616  022714  140101                  CMP     #140101,(R4)            ;SEE IF EXPECTED RESULT
3358 012622  001003                          BNE     1$                      ;BRANCH IF BAD
3359 012624  000337  000400                  SWAB    @#400                   ;400=101 300
3360 012630  100401                          BMI     2$                      ;BRANCH IF GOOD
3361                                                                         ;ERROR! BAD SWAB MODE 37
3362 012632  104001                  1$:     ERROR   .1              ;CPU ERROR
3363 012634                          2$:
3364
3365                                 ;
3368 012634                          MRR0:
3369
```

C6

********** DOUBLE OPERAND TESTS **********

```
3370                                    ;       TEST ROR MODE 0
3371 01263  012704  052525              MOV     #52525,R4               ;R4=52525
3372 01264  000257                      CCC                             ;CC=0000
3373 012642 006004                      ROR     R4                      ;R4 SHOULD = 25252 WITH CARRY
3374 012644 103003                      BCC     1$                      ;ERROR IF NO CARRY
3375 012646 022704  025252              CMP     #25252,R4               ;SEE IF R4= EXPECTED
3376 012652 001401                      BEQ     2$                      ;BRANCH IF GOOD
3377                                                                    ;ROR MODE 0 FAILED
3378 012654 104001                  1$: ERROR   .1              ;CPU ERROR
3379 012656                         2$:
3380
3381                                    ;
3384 012656                         MRRB1:
3385
3386                                    ;       TEST RORB MODE 1
3387 012656 005004                      CLR     R4                      ;R4=0
3388 012660 012714  000001              MOV     #1,(R4)                 ;0=1
3389 012664 000277                      SCC                             ;CC=1111
3390 012666 106014                      RORB    (R4)                    ;0=000200, NO C
3391 012670 103004                      BCC     1$                      ;ERROR IF NO CARRY
3392 012672 100003                      BPL     1$                      ;ERROR IF PLUS
3393 012674 022714  000200              CMP     #200,(R4)               ;CHECK RESULT
3394 012700 001401                      BEQ     2$                      ;BRANCH IF GOOD
3395 012702 104001                  1$: ERROR   .1              ;CPU ERROR
3396                                                            ;BAD RESULT
3397 012704                         2$:
3398
3399                                    ;
3401
3403 012704                         MJ:
3404
3405                                    ;       TEST JMP - ALL MODES
3406 012704 012737  000001  003072      MOV     #1,@#SEQ                ;SETUP TEST SEQUENCER
3407 012712 012701  012756              MOV     #MJU1,R1                ;SET MODE 1 JUMP ADDRESS
3408 012716 000111                      JMP     (R1)                    ;*JMP MODE 1
3409 012720 023727  003072  000002  MJU2: CMP     @#SEQ,#2                ;CHECK FOR CORRECT SEQUENCE
3410 012726 001401                      BEQ     MJU2A                   ;BRANCH IF GOOD
3411                                                                    ;MODE 2 JUMP FAILED
3412 012730 104001                  1$: ERROR   .1              ;CPU ERROR
3413 012732 020127  012722          MJU2A: CMP  R1,#MJU2+2              ;CHECK FOR AUTO-INCREMENT
3414 012736 001401                      BEQ     MJU2B                   ;BRANCH IF GOOD
3415 012740 104001                  2$: ERROR   .1              ;CPU ERROR
3416                                                            ;AUTO-INCR FAILED
3417 012742 005237  003072          MJU2B: INC  @#SEQ                   ;UPDATE TEST SEQUENCER
3418 012746 012701  012754              MOV     #MJ2,R1                 ;SETUP MODE 3 JUMP
3419 012752 000131                      JMP     @(R1)+                  ;JUMP MODE 3
3420 012754 013002                  MJ2:    .WORD   MJU3                ;MODED 3 DETINATION
3421 012756 023727  003072  000001  MJU1: CMP     @#SEQ,#1                ; TEST FOR CORRECT SEQUENCE
3422 012764 001401                      BEQ     MJU1A                   ;BRANCH IF GOOD
3423 012766 104001                  3$: ERROR   .1              ;CPU ERROR
3424                                                            ;JMP OUT OF SEQUENCE
3425 012770 005237  003072          MJU1A: INC  @#SEQ                   ;UPDATE SEQUENCE
3426 012774 012701  012720              MOV     #MJU2,R1                ;SETUP MODE 2 DESTINATION
Z    3427 013000 000121                 JMP     (R1)+                   ;JUMP MODE 2
3428 013002 023727  003072  000003  MJU3: CMP     @#SEQ,#3                ; TEST FOR CORRECT SEQUENCE
3429 013010 001401                      BEQ     MJU3A                   ;BRANCH IF GOOD
3430 013012 104001                  4$: ERROR   .1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS **********

3431
3432 013014  022701  012756          MJU3A:  CMP     @MJ2+2,R1       ;JMP OUT OF SEQUENCE
3433 013020  001401                         BEQ     MJU3B                   ; TEST AUTO-INCREMENT
3434 013022  104001                  5$:    ERROR   +1                      ;BRANCH IF GOOD
3435                                                                 ;CPU ERROR
3436 013024  005237  003072          MJU3B:  INC     @#SEQ           ;AUTO-INCREMENT FAILED MODE 3
3437 013030  012701  013100                  MOV     #MJU4+2,R1              ;UPDATE SEQUENCER
3438 013034  000141                         JMP     -(R1)                   ;SETUP DESTINATION MODE 4
3439 013036  000000          MJU5:          HALT                            ;EXECUTE JUMP MODE 4
3440 013040  022701  013134                  CMP     #MJ5,R1         ;CHECK AUTO-DECREMENT
3441 013044  001401                         BEQ     MJU5A                   ;BRANCH IF GOOD AUTO-DEC
3442 013046  104001                  6$:    ERROR   +1              ;CPU ERROR
3443                                                                 ;AUTO-DEC FAIELD MODE 5
3444 013050  023727  003072  000005 MJU5A:  CMP     @#SEQ,#5                ; TEST CORRECT SEQUENCE
3445 013056  001401                         BEQ     MJU5B                   ;BRANCH IF GOOD SEQUENCE
3446 013060  104001                  7$:    ERROR   +1              ;CPU ERROR
3447                                                                 ;JMP OUT OF SEQUENCE
3448 013062  005237  003072          MJU5B:  INC     @#SEQ                   ;UPDATE SEQUENCE COUNT
3449 013066  012701  013131                  MOV     #MJU6-5,R1              ;SETUP DESTINATION MODE6
3450 013072  000161  000005                 JMP     +5(R1)                  ;JUMP MODE 6
3451                                  ;;
3452 013076  000240          MJU4:          NOP
3453 013100  022701  013076                  CMP     #MJU4,R1                ; TEST AUTO-DECR
3454 013104  001401                         BEQ     MJU4A                   ;BRANCH IF GOOD
3455 013106  104001                  8$:    ERROR   +1              ;CPU ERROR
3456                                                                 ;MODE 4 AUTO-DEC FAILED
3457 013110  023727  003072  000004 MJU4A:  CMP     @#SEQ,#4                ; TEST FOR CORRECT SEQUENCE
3458 013116  001401                         BEQ     MJU4B                   ;BRANCH IF CORRECT SEQUENCE
3459 013120  104001                  9$:    ERROR   +1              ;CPU ERROR
3460                                                                 ;INCORRECT JMP SEQUENCE
3461 013122  005237  003072          MJU4B:  INC     @#SEQ                   ;UPDATE SEQUENCE
3462 013126  012701  013136                  MOV     #MJ5+2,R1               ;SETUP MODE 5 POINTER
3463 013132  000151                         JMP     @-(R1)                  ;EXECUTE MODE 5 JMP
3464                                  ;
3465 013134  013040          MJ5:           .WORD   MJU5+2                  ;POINTER MODE 5
3466                                  ;
3467 013136  022737  000006  003072 MJU6:   CMP     #6,@#SEQ                ;CHECK FOR CORRECT SEQUENCE
3468 013144  001401                         BEQ     MJU6A                   ;BRANCH IF GOOD
3469 013146  104001                  10$:   ERROR   +1              ;CPU ERROR
3470                                                                 ;INCORRECT SEQUENCE
3471 013150  005237  003072          MJU6A:  INC     @#SEQ                   ;UPDATE SEQUENCER
3472 013154  012701  013174                  MOV     #MJ7+10,R1              ;SETUP INDEX
3473 013160  000171  177770                 JMP     @-10(R1)                ;EXECUTE MODE 7 JUMP
3474 013164  013170          MJ7:           .WORD   MJU7                    ;POINTER FOR MODE 7
3475 013166  000000                         HALT
3476 013170  022737  000007  003072 MJU7:   CMP     #7,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3477 013176  001401                         BEQ     MJU7E                   ;BRANCH IF GOOD SEQUENCE
3478 013200  104001                  11$:   ERROR   +1              ;CPU ERROR
3479                                                                 ; TESTING OUT OF SEQUENCE
3480 013202          MJU7E:
3481                                  ;
3492
3493                                  ;       TEST THAT PRE-FETCH BUFFER CAN BE OVER WRITTEN
3494 013202  012701  013512                  MOV     #128$,R1                ;SET UP R1 WITH ADDRESS OF ERROR
3495                                                                         ;ROUTINE
3496         000040                  .REPT   32.
3498 013206  012717                         MOV     (PC)+,(PC)              ;WRITE THE NOP OVER THE JMP INSTRUCTION
```

E6

********** DOUBLE OPERAND TESTS **********

```
013210  000240                  .WORD   NOP             ;NOP INSTRUCTION
013212  000111          64$:    .WORD   111             ;JMP    (R1)
013214  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013216  000240                  .WORD   NOP             ;NOP INSTRUCTION
013220  000111          65$:    .WORD   111             ;JMP    (R1)
013222  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013224  000240                  .WORD   NOP             ;NOP INSTRUCTION
013226  000111          66$:    .WORD   111             ;JMP    (R1)
013230  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013232  000240                  .WORD   NOP             ;NOP INSTRUCTION
013234  000111          67$:    .WORD   111             ;JMP    (R1)
013236  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013240  000240                  .WORD   NOP             ;NOP INSTRUCTION
013242  000111          68$:    .WORD   111             ;JMP    (R1)
013244  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013246  000240                  .WORD   NOP             ;NOP INSTRUCTION
013250  000111          69$:    .WORD   111             ;JMP    (R1)
013252  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013254  000240                  .WORD   NOP             ;NOP INSTRUCTION
013256  000111          70$:    .WORD   111             ;JMP    (R1)
013260  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013262  000240                  .WORD   NOP             ;NOP INSTRUCTION
013264  000111          71$:    .WORD   111             ;JMP    (R1)
013266  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013270  000240                  .WORD   NOP             ;NOP INSTRUCTION
013272  000111          72$:    .WORD   111             ;JMP    (R1)
013274  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013276  000240                  .WORD   NOP             ;NOP INSTRUCTION
013300  000111          73$:    .WORD   111             ;JMP    (R1)
013302  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013304  000240                  .WORD   NOP             ;NOP INSTRUCTION
013306  000111          74$:    .WORD   111             ;JMP    (R1)
013310  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013312  000240                  .WORD   NOP             ;NOP INSTRUCTION
013314  000111          75$:    .WORD   111             ;JMP    (R1)
013316  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013320  000240                  .WORD   NOP             ;NOP INSTRUCTION
013322  000111          76$:    .WORD   111             ;JMP    (R1)
013324  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013326  000240                  .WORD   NOP             ;NOP INSTRUCTION
013330  000111          77$:    .WORD   111             ;JMP    (R1)
013332  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013334  000240                  .WORD   NOP             ;NOP INSTRUCTION
013336  000111          78$:    .WORD   111             ;JMP    (R1)
013340  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013342  000240                  .WORD   NOP             ;NOP INSTRUCTION
013344  000111          79$:    .WORD   111             ;JMP    (R1)
013346  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013350  000240                  .WORD   NOP             ;NOP INSTRUCTION
013352  000111          80$:    .WORD   111             ;JMP    (R1)
013354  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013356  000240                  .WORD   NOP             ;NOP INSTRUCTION
013360  000111          81$:    .WORD   111             ;JMP    (R1)
013362  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013364  000240                  .WORD   NOP             ;NOP INSTRUCTION
013366  000111          82$:    .WORD   111             ;JMP    (R1)
013370  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
```

```
********** DOUBLE OPERAND TESTS **********

013372  000240                          .WORD   NOP             ;NOP INSTRUCTION
013374  000111                  83$:    .WORD   111             ;JMP    (R1)
013376  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013400  000240                          .WORD   NOP             ;NOP INSTRUCTION
013402  000111                  84$:    .WORD   111             ;JMP    (R1)
013404  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013406  000240                          .WORD   NOP             ;NOP INSTRUCTION
013410  000111                  85$:    .WORD   111             ;JMP    (R1)
013412  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013414  000240                          .WORD   NOP             ;NOP INSTRUCTION
013416  000111                  86$:    .WORD   111             ;JMP    (R1)
013420  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013422  000240                          .WORD   NOP             ;NOP INSTRUCTION
013424  000111                  87$:    .WORD   111             ;JMP    (R1)
013426  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013430  000240                          .WORD   NOP             ;NOP INSTRUCTION
013432  000111                  88$:    .WORD   111             ;JMP    (R1)
013434  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013436  000240                          .WORD   NOP             ;NOP INSTRUCTION
013440  000111                  89$:    .WORD   111             ;JMP    (R1)
013442  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013444  000240                          .WORD   NOP             ;NOP INSTRUCTION
013446  000111                  90$:    .WORD   111             ;JMP    (R1)
013450  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013452  000240                          .WORD   NOP             ;NOP INSTRUCTION
013454  000111                  91$:    .WORD   111             ;JMP    (R1)
013456  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013460  000240                          .WORD   NOP             ;NOP INSTRUCTION
013462  000111                  92$:    .WORD   111             ;JMP    (R1)
013464  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013466  000240                          .WORD   NOP             ;NOP INSTRUCTION
013470  000111                  93$:    .WORD   111             ;JMP    (R1)
013472  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013474  000240                          .WORD   NOP             ;NOP INSTRUCTION
013476  000111                  94$:    .WORD   111             ;JMP    (R1)
013500  012717                          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013502  000240                          .WORD   NOP             ;NOP INSTRUCTION
013504  000111                  95$:    .WORD   111             ;JMP    (R1)
3499 013506  000137  013514            JMP     @#129$          ;JUMP OVER ERROR CALL
3500 013512                     128$:                           ;ERROR! PRE-FETCH BUFFER WAS NOT
3501                                                            ;OVER WRITTEN
3502 013512  104001                    ERROR   +1              ;CPU ERROR
3503                             ;
3504                             ;      NOW RESTORE THE OVER WRITTEN JMP INSTRUCTIONS FOR THE NEXT PASS.
3505                             ;
3506 013514  012702  000040      129$:  MOV     #32.,R2          ;SET UP R2 AS COUNTER
3507 013520  012703  013212            MOV     #64$,R3          ;SET UP R3 AS POINTER
3508 013524  012713  000111      130$:  MOV     #111,(R3)        ;RESTORE OVER WRITTEN JUMPS
3509 013530  062703  000006            ADD     #6,R3            ;POINT TO NEXT OVER WRITTEN ADDR.
3510 013534  077205                    SOB     R2,130$          ;DO UNTIL R2=0
3511
3513                             ;
3515 013536                     MJP:
3516
3517                             ;      TEST JMP MODES 17,27,37,67,77
3518 013536  012737  000000  003072    MOV     #0,@#SEQ         ;SETUP TEST SEQUNCER
3519 013544  000117                    JMP     (R7)             ;JUMP MODE 17(SHOULD BE IN-LINE)
```

```
                                        ;
013546  005737  003072                  MJP17:  TST     @#SEQ                   ;CHECK SEQUENCE
013552  001401                                  BEQ     2$                      ;BRANCH IF GOOD
                                                                                ;ERROR! BAD JUMP
013554  104001                          1$:     ERROR   +1              ;CPU ERROR
013556  005237  003072                  2$:     INC     @#SEQ                   ;UPDATE SEQUENCE NUMBER
Z 013562  000127  000401                        JMP     #401                    ;JUMP MODE 27
                                                                                ;(THE #401=BR UPDATED PC+2)
                                        ;
013566  000000                                  HALT                            ;
                                        ;
013570  023727  003072  000001          MJP27:  CMP     @#SEQ,#1                ;CHECK IF CORRECT SEQUENCE
013576  001401                                  BEQ     MJP27A                  ;BRANCH IF IN SEQUENCE
013600  104001                          1$:     ERROR   +1              ;CPU ERROR
                                                                        ; TEST OUT OF SEQUENCE
013602  005237  003072                  MJP27A: INC     @#SEQ                   ;UPDATE SEQUENCER
013606  000137  013654                          JMP     @#MJP37                 ;JUMP MODE 37
013612  023727  003072  000003          MJP67:  CMP     @#SEQ,#3                ;CHECK FOR CORRECT SEQUENCE
013620  001401                                  BEQ     MJP67A                  ;BRANCH IF IN SEQUENCE
013622  104001                          2$:     ERROR   +1              ;CPU ERROR
                                                                        ; TEST OUT OF SEQUENCE
013624  005237  003072                  MJP67A: INC     @#SEQ                   ;UPDATE SEQUENCER
013630  000257                                  CCC                             ;INSURE ZBIT=0
013632  000177  000002                          JMP     @MJP67B                 ;JUMP MODE 7
                                        ;
013636  000000                                  HALT                            ;
                                        ;
013640  013642                          MJP67B: .WORD   MJP77                   ;
                                        ;
013642  023727  003072  000004          MJP77:  CMP     @#SEQ,#4                ; TEST FOR CORRECT SEQUENCE
013650  001412                                  BEQ     MJP77E                  ;BRANCH IF IN SEQUENCE
013652  104001                          3$:     ERROR   +1              ;CPU ERROR
                                                                        ; TEST OUT OF SEQUENCE
013654  023727  003072  000002          MJP37:  CMP     @#SEQ,#2                ; TEST FOR CORRECT SEQUENCE
013662  001401                                  BEQ     MJP37A                  ;BRANCH IF IN SEQUENCE
013664  104001                          4$:     ERROR   +1              ;CPU ERROR
                                                                        ; TEST OUT OF SEQUENCE
013666  005237  003072                  MJP37A: INC     @#SEQ                   ;UPDATE SEQUENCER
013672  000167  177714                          JMP     MJP67                   ;JUMP MODE 6
                                        ;
                                        ;
013676                                  MJP77E:
                                        ;
                                        ;
013676                                  MJSR:

                                        ;       TEST JSR ALL MODES
013676  010637  003074                          MOV     R6,@#SPS                ;SAVE STACK POINTER LOCATION
013702  010637  003076                          MOV     R6,@#SPSJ               ;
013706  162737  000002  003076                  SUB     #2,@#SPSJ               ;SPSJ = R6 AFTER DECRIMENT
013714  012737  000001  003072                  MOV     #1,@#SEQ                ;SETUP SEQUENCE COUNTER
013722  012701  014016                          MOV     #MJSR1,R1               ;SETUP INITIAL JUMP IN MODE 1
013726  005004                                  CLR     R4                      ;
013730  005104                                  COM     R4                      ;R4=-1 TO BE SAVED ON STACK
013732  004411                                  JSR     R4,(R1)                 ;JSR MODE 1
                                        ;
013734  022737  000002  003072          MJSR2:  CMP     #2,@#SEQ                ; TEST FOR CORRECT SEQUENCE
```

```
********** DOUBLE OPERAND TESTS ******************

 3579 013742  001401                      BEQ     MJSR2A          ;BRANCH IF GOOD
 3580 013744  104001            5$:       ERROR   +1              ;CPU ERROR
 3581                                                             ;MODE 2 JUMPED TO OUT OF SEQUENCE
 3582 013746  023706  003076    MJSR2A:   CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3583 013752  001006                      BNE     6$              ;BRANCH IF STACK INCORRECT
 3584 013754  021627  125252              CMP     (R6),#125252    ;VERIFY CONTENTS OF STACK
 3585 013760  001003                      BNE     6$              ;BRANCH IF DATA ON STACK INCORRECT
 3586 013762  022704  014076              CMP     #MJSR4,R4       ;SEE IF CORRECT RETURN ADDRESS
 3587 013766  001401                      BEQ     MJSR2B          ;BRANCH IF GOOD
 3588 013770  104001            6$:       ERROR   +1              ;CPU ERROR
 3589                                                             ;JSR MODE 2 FAILED
 3590 013772  005237  003072    MJSR2B:   INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3591 013776  013706  003074              MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3592 014002  012701  014014              MOV     #MJSRA,R1       ;SETUP JSR MODE 3
 3593 014006  005004                      CLR     R4              ;DIFFERENT DATA TO R4
 3594 014010  004431    .                 JSR     R4,@(R1)+       ;JSR MODE 3
 3595 014012  000000                      HALT
 3596 014014  014162            MJSRA:    .WORD   MJSR3           ;LITERAL FOR JUMP MODE 3
 3597                                    ;
 3598 014016  022737  000001  003072  MJSR1:   CMP     #1,@#SEQ        ; TEST FOR CORRECT SEQUENCE
 3599 014024  001401                      BEQ     MJSR1A          ;BRANCH IF GOOD
 3600 014026  104001            7$:       ERROR   +1              ;CPU ERROR
 3601                                                             ;MODE 1 JUMPED TO OUT OF SEQUENCE
 3602 014030  023706  003076    MJSR1A:   CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3603 014034  001006                      BNE     8$              ;BRANCH IF STACK INCORRECT
 3604 014036  021627  177777              CMP     (R6),#-1        ;VERIFY CONTENT OF STACK
 3605 014042  001003                      BNE     8$              ;BRANCH IF DATA ON STACK INCORRECT
 3606 014044  022704  013734              CMP     #MJSR2,R4       ;SEE IF CORRECT RETURN ADDRESS
 3607 014050  001401                      BEQ     MJSR1B          ;BRANCH IF GOOD
 3608 014052  104001            8$:       ERROR   +1              ;CPU ERROR
 3609                                                             ;JSR MODE 2 FAILED
 3610 014054  005237  003072    MJSR1B:   INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3611 014060  013706  003074              MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3612 014064  012704  125252              MOV     #125252,R4      ;SETUP R4 DATA
 3613 014070  012701  013734              MOV     #MJSR2,R1       ;SETUP MODE 2 JUMP ADDRESS
Z3614 014074  004421                      JSR     R4,(R1)+        ;+JUMP MODE 2
 3615                                    ;
 3616                                    ;
 3617 014076  022737  000004  003072  MJSR4:   CMP     #4,@#SEQ        ; TEST FOR CORRECT SEQUENCE
 3618 014104  001401                      BEQ     MJSR4A          ;BRANCH IF GOOD
 3619 014106  104001            9$:       ERROR   +1              ;CPU ERROR
 3620                                                             ;MODE 4 JUMPED TO OUT OF SEQUENCE
 3621 014110  023706  003076    MJSR4A:   CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3622 014114  001006                      BNE     10$             ;BRANCH IF STACK INCORRECT
 3623 014116  021627  052525              CMP     (R6),#052525    ;VERIFY CONTENTS OF STAACK
 3624 014122  001003                      BNE     10$             ;BRANCH IF DATA ON STACK INCORRECT
 3625 014124  022704  014244              CMP     #MJSR6,R4       ;SEE IF CORRECT RETURN ADDRESS
 3626 014130  001401                      BEQ     MJSR4B          ;BRANCH IF GOOD
 3627 014132  104001            10$:      ERROR   +1              ;CPU ERROR
 3628                                                             ;JSR MODE 4 FAILED
 3629 014134  005237  003072    MJSR4B:   INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3630 014140  013706  003074              MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3631 014144  012704  000377              MOV     #377,R4         ;SETUP R4 DATA
 3632 014150  012701  014162              MOV     #MJSR8+2,R1     ;SETUP JSR VECTOR
 3633 014154  004451                      JSR     R4,@-(R1)       ;JSR MODE 5
 3634 014156  000000                      HALT
 3635 014160  014330            MJSR8:    .WORD   MJSR5           ;MODE 5 VECTOR
```

********** DOUBLE OPERAND TESTS **********

```
3636                                    ;
3637                                    ;
3638 014162  022737  000003  003072  MJSR3:  CMP     #3,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3639 014170  001401                          BEQ     MJSR3A                  ;BRANCH IF GOOD
3640 014172  104001                  11$:    ERROR   +1              ;CPU ERROR
3641                                                                 ;MODE 3 JUMPED TO OUT OF SEQUENCE
3642 014174  023706  003076          MJSR3A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3643 014200  001006                          BNE     12$                     ;BRANCH IF STACK INCORRECT
3644 014202  021627  000000                  CMP     (R6),#0                 ;VERIFY CONTENTS OF STAACK
3645 014206  001003                          BNE     12$                     ;BRANCH IF DATA ON STACK INCORRECT
3646 014210  022704  014012                  CMP     #MJSRA-2,R4             ;SEE IF CORRECT RETURN ADDRESS
3647 014214  001401                          BEQ     MJSR3B                  ;BRANCH IF GOOD
3648 014216  104001                  12$:    ERROR   +1              ;CPU ERROR
3649                                                                 ;JSR MODE 3 FAILED
3650 014220  005237  003072          MJSR3B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3651 014224  013706  003074                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3652 014230  012704  052525                  MOV     #052525,R4              ;SETUP R4 DATA
3653 014234  012701  014100                  MOV     #MJSR4+2,R1             ;SETUP JSR VECTOR
3654 014240  000257                          CCC                             ;CLEAR CONDITION CODES
3655 014242  004441                          JSR     R4,-(R1)                        ;JSR MODE 4
3656                                    ;
3657                                    ;
3658 014244  022737  000006  003072  MJSR6:  CMP     #6,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3659 014252  001401                          BEQ     MJSR6A                  ;BRANCH IF GOOD
3660 014254  104001                  13$:    ERROR   +1              ;CPU ERROR
3661                                                                 ;MODE 6 JUMPED TO OUT OF SEQUENCE
3662 014256  023706  003076          MJSR6A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3663 014262  001006                          BNE     14$                     ;BRANCH IF STACK INCORRECT
3664 014264  021627  123456                  CMP     (R6),#123456            ;VERIFY CONTENTS OF STACK
3665 014270  001003                          BNE     14$                     ;BRANCH IF DATA ON STACK INCORRECT
3666 014272  022704  014412                  CMP     #MJSR7,R4               ;SEE IF CORRECT RETURN ADDRESS
3667 014276  001401                          BEQ     MJSR6B                  ;BRANCH IF GOOD
3668 014300  104001                  14$:    ERROR   +1              ;CPU ERROR
3669                                                                 ;JSR MODE 6 FAILED
3670 014302  005237  003072          MJSR6B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3671 014306  013706  003074                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3672 014312  012704  177773                  MOV     #-5,R4                  ;SETUP R4 DATA
3673 014316  012701  014336                  MOV     #MJSRC+10,R1            ;SETUP JSR VECTOR
3674 014322  004471  177770                  JSR     R4,@-10(R1)             ;JSR MODE 7
3675 014326  014412                  MJSRC:  .WORD   MJSR7                   ;JSR VECTOR
3676                                    ;
3677                                    ;
3678 014330  022737  000005  003072  MJSR5:  CMP     #5,@#SEQ                ; TEST FOR CORRECT SEQUENCE
3679 014336  001401                          BEQ     MJSR5A                  ;BRANCH IF GOOD
3680 014340  104001                  15$:    ERROR   +1              ;CPU ERROR
3681                                                                 ;MODE 5 JUMPED TO OUT OF SEQUENCE
3682 014342  023706  003076          MJSR5A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
3683 014346  001006                          BNE     16$                     ;BRANCH IF STACK INCORRECT
3684 014350  021627  000377                  CMP     (R6),#377               ;VERIFY CONTENTS OF STACK
3685 014354  001003                          BNE     16$                     ;BRANCH IF DATA ON STACK INCORRECT
3686 014356  022704  014156                  CMP     #MJSR8-2,R4             ;SEE IF CORRECT RETURN ADDRESS
3687 014362  001401                          BEQ     MJSR5B                  ;BRANCH IF GOOD
3688 014364  104001                  16$:    ERROR   +1              ;CPU ERROR
3689                                                                 ;JSR MODE 5 FAILED
3690 014366  005237  003072          MJSR5B: INC     @#SEQ                   ;UPDATE SEQUENCE COUNTER
3691 014372  013706  003074                  MOV     @#SPS,R6                ;RELOAD STACK POINTER
3692 014376  012704  123456                  MOV     #123456,R4              ;SETUP DATA IN R4
```

```
********** DOUBLE OPERAND TESTS ******************

 3693 014402  012701  014254                 MOV     #MJSR6+10,R1            ;SETUP JSR VECTOR
 3694 014406  004461  177770                 JSR     R4,-10(R1)              ;JUMP MODE 6
 3695                              ;
 3696                              ;
 3697 014412  022737  000007  003072 MJSR7:  CMP     #7,@#SEQ                ; TEST FOR CORRECT SEQUENCE
 3698 014420  001401                         BEQ     MJSR7A                  ;BRANCH IF GOOD
 3699 014422  104001                 17$:    ERROR   +1              ;CPU ERROR
 3700                                                                 ;MODE 7 JUMPED TO OUT OF SEQUENCE
 3701 014424  023706  003076         MJSR7A: CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
 3702 014430  001006                         BNE     18$                     ;BRANCH IF STACK INCORRECT
 3703 014432  021627  177773                 CMP     (R6),#-5                ;VERIFY CONTENTS OF STAACK
 3704 014436  001003                         BNE     18$                     ;BRANCH IF DATA ON STACK INCORRECT
 3705 014440  022704  014326                 CMP     #MJSR5-2,R4             ;SEE IF CORRECT RETURN ADDRESS
 3706 014444  001401                         BEQ     MJSR7E                  ;BRANCH IF GOOD
 3707 014446  104001                 18$:    ERROR   +1              ;CPU ERROR
 3708                                                                 ;JSR MODE 7 FAILED
 3709 014450                         MJSR7E:
 3710 014450  013706  003074                 MOV     @#SPS,R6                ;REPLACE STACK
 3711
 3712                              ;
 3715 014454                         MJRA:
 3716
 3717                              ;       TEST JSR MODES 27, 37, 67, 77
 3718 014454  012737  000001  003072         MOV     #1,@#SEQ                ;SETUP SEQUENCER
 3719 014462  010637  003074                 MOV     R6,@#SPS                ;SAVE STACK ADDRESS
 3720 014466  010637  003076                 MOV     R6,@#SPSJ               ;SAVE STACK DECRIMENT ADDRESS
 3721 014472  162737  000002  003076         SUB     #2,@#SPSJ               ;
 3722 014500  012704  177777                 MOV     #-1,R4                  ;SETUP R4 DATA
Z 3723 014504  004427  000240                JSR     R4,#240                 ;EXECUTE A JSR MODE 27
 3724                              ;
 3725 014510  022737  000001  003072 MJR27:  CMP     #1,@#SEQ                ;VERIFY COERRECT TEST SEQUENCE
 3726 014516  001011                         BNE     1$                      ;INCORRECT TEST SEQUENCE
 3727 014520  023706  003076                 CMP     @#SPSJ,R6               ;VERIFY STACK POINTER
 3728 014524  001006                         BNE     1$
 3729 014526  021627  177777                 CMP     (R6),#-1                ;VERIFY R4 GOT LOADED ON THE STACK
 3730 014532  001003                         BNE     1$                      ;BRANCH IF INCORRECT STACK CONTENTS
 3731 014534  020427  014510                 CMP     R4,#MJR27               ;VERIFY CORRECT RETURN ADDRESS
 3732 014540  001401                         BEQ     MJR27A                  ;BRANCH IF GOOD RETURN ADDRESS ON STACK
 3733 014542  104001                 1$:     ERROR   +1              ;CPU ERROR
 3734                                                                 ;MODE 27 FAILED
 3735 014544  005237  003072         MJR27A: INC     @#SEQ                   ;UPDATE SEQUENCER
 3736 014550  012704  152525                 MOV     #152525,R4              ;SETUP R4 TEST DATA
 3737 014554  013706  003074                 MOV     @#SPS,R6                ;RESET STACK POINTER
 3738 014560  004437  014646                 JSR     R4,@#MJR37              ;JSR MODE 37
 3739 014564  000000                 MJR27B: HALT
 3740                              ;
 3741 014566  023727  003072  000003 MJR67:  CMP     @#SEQ,#3                ;VERIFY TEST SEQUENCE
 3742 014574  001011                         BNE     2$                      ;INCORRECT TEST SEQUENCE
 3743 014576  023706  003076                 CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
 3744 014602  001006                         BNE     2$                      ;INCORRECT STACK DECRIMENT
 3745 014604  021627  000125                 CMP     (R6),#125               ;VERFIFY STACK WAS LOADED
 3746 014610  001003                         BNE     2$
 3747 014612  020427  014722                 CMP     R4,#MJR77               ;VERIFY RETURN ADDRESS
 3748 014616  001401                         BEQ     MJR67A                  ;BRANCH IF GOOD
 3749 014620  104001                 2$:     ERROR   +1              ;CPU ERROR
 3750                                                                 ;MODE 67 FAILED
 3751 014622  005237  003072         MJR67A: INC     @#SEQ                   ;UPDATE SEQUENCER
```

K6


```
014626  013706  003074                  MOV     @#SPS,R6                ;RESET STACK
014632  012704  000001                  MOV     #1,R4                   ;SETUP R4 DATA
014636  004477  000002                  JSR     R4,@MJR6B               ;JSR MODE 77
014642  000000                  MJR6A:  HALT
014644  014722                  MJR6B:  .WORD   MJR77                   ;DATA FOR MODE 77 JUMP
                                ;
014646  023727  003072  000002  MJR37:  CMP     @#SEQ,#2                ;VERIFY TEST SEQUENCE
014654  001011                          BNE     2$                      ;INCORRECT TEST SEQUENCE
014656  023706  003076                  CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
014662  001006                          BNE     2$                      ;INCORRECT STACK DECRIMENT
014664  021627  152525                  CMP     (R6),#152525            ;VERFIFY STACK WAS LOADED
014670  001003                          BNE     2$
014672  020427  014564                  CMP     R4,#MJR27B              ;VERIFY RETURN ADDRESS
014676  001401                          BEQ     MJR37A                  ;BRANCH IF GOOD
014700  104001                  2$:     ERROR   +1                  ;CPU ERROR
                                                                    ;MODE 37 FAILED
014702  005237  003072          MJR37A: INC     @#SEQ                   ;UPDATE SEQUENCER
014706  013706  003074                  MOV     @#SPS,R6                ;RELOAD STACK
014712  012704  000125                  MOV     #125,R4                 ;SETUP R4 TEST DATA
014716  004467  177644                  JSR     R4,MJR67                ;JSR MODE 6
                                ;
014722  023727  003072  000004  MJR77:  CMP     @#SEQ,#4                ;VERIFY TEST SEQUENCE
014730  001011                          BNE     2$                      ;INCORRECT TEST SEQUENCE
014732  023706  003076                  CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
014736  001006                          BNE     2$                      ;INCORRECT STACK DECRIMENT
014740  021627  000001                  CMP     (R6),#1                 ;VERFIFY STACK WAS LOADED
014744  001003                          BNE     2$
014746  020427  014642                  CMP     R4,#MJR6A               ;VERIFY RETURN ADDRESS
014752  001401                          BEQ     MJR77A                  ;BRANCH IF GOOD
014754  104001                  2$:     ERROR   +1                  ;CPU ERROR
                                                                    ;MODE 77 FAILED
014756                          MJR77A:
                                ;
014756  013706  003074                  MOV     @#SPS,R6                ;RESET STACK


                                ;
014762                          MRTS:


                                ;       TEST RTS AND RTS R6
014762  012706  001000                  MOV     #STBOT,R6               ;INSURE VALID STACK
014766  012746  123456                  MOV     #123456,-(R6)           ;SETUP TEST REGISTER
014772  012703  015002                  MOV     #RTS1,R3                ;SETUP TEST PC
014776  000203                          RTS     R3                      ;**TEST INSTRUCTION
015000  104001                          ERROR   +1                  ;CPU ERROR
                                                                    ;INCORRECT PC ON RTS
015002  022703  123456          RTS1:   CMP     #123456,R3
015006  001401                          BEQ     RTS6                    ;BRANCH IF GOOD
015010  104001                          ERROR   +1                  ;CPU ERROR
                                                                    ;REGISTER CONTENTS INCORRECT
                                        ;
                                        ;THIS TEST CHECKS AN UN-TESTED PLA TERM
                                        ;
015012  010601                  RTS6:   MOV     R6,R1                   ;SAVE STACK IN R1
015014  012705  015026                  MOV     #1$,R5                  ;MOVE EXPECTED RETURN ADDR TO R5
015020  010506                          MOV     R5,R6                   ;MOVE RETURN ADDR TO R6
015022  000206                          RTS     R6                      ;
015024  104001                          ERROR   +1                  ;CPU ERROR
```

********** DOUBLE OPERAND TESTS **********

```
3811                                                                    ;ERROR! RTS NOT EXECUTED
3812 015026  021506                    1$:     CMP     (R5),R6                 ;IS R6=315067
3813 015030  001401                            BEQ     RTSE                    ;IF IT IS THEN GO TO END OF TEST
3814 015032  104001                            ERROR   .1              ;CPU ERROR
3815                                                                    ;ERROR! WRONG ADDR IN R6
3816 015034  010106                    RTSE:   MOV     R1,R6                   ;RESTORE STACK
3817
3820 015036                            TSMMU0:
3821                                   ;       SETUP AND TEST KERNEL, SUPERVISOR AND USER STACKS
3822 015036  012737  040000  177776            MOV     #40000,@#177776         ;SET PS TO SUP MODE
3823 015044  012706  177777                    MOV     #177777,R6              ;INIT SUP STACK TO ALL ONES
3824 015050  022706  177777                    CMP     #177777,R6              ;ARE ALL BITS SET
3825 015054  001401                            BEQ     1$                      ;YES GO ON
3826                                                                    ;NO GO TO ERROR
3827 015056  104001                            ERROR   .1              ;CPU ERROR
3828 015060  005006                    1$:     CLR     R6                      ;SET SUP STACK TO ALL ZEROES
3829 015062  022706  000000                    CMP     #0,R6                   ;ARE ALL BITS CLEARED
3830 015066  001401                            BEQ     2$                      ;YES GO ON
3831                                                                    ;NO GO TO ERROR
3832 015070  104001                            ERROR   .1              ;CPU ERROR
3833 015072  012706  125252            2$:     MOV     #125252,R6              ;SET SUP STACK TO ALTERNATING PATTERN
3834 015076  022706  125252                    CMP     #125252,R6              ;IS SUP SP CORRECT
3835 015102  001401                            BEQ     3$                      ;YES GO ON
3836                                                                    ;NO GO TO ERROR
3837 015104  104001                            ERROR   .1              ;CPU ERROR
3838 015106  012706  052525            3$:     MOV     #52525,R6               ;SET SUP STACK TO ALTERNATING PATTERN
3839 015112  022706  052525                    CMP     #52525,R6               ;IS SUP SP CORRECT
3840 015116  001401                            BEQ     4$                      ;YES GO ON
3841                                                                    ;NO GO TO ERROR
3842 015120  104001                            ERROR   .1              ;CPU ERROR
3843 015122  012706  000700            4$:     MOV     #700,R6                 ;SETUP SUP SP
3844 015126  012737  140000  177776            MOV     #140000,@#177776        ;SET PS TO USER MODE
3845 015134  012706  177777                    MOV     #177777,R6              ;INIT USER STACK TO ALL ONES
3846 015140  022706  177777                    CMP     #177777,R6              ;ARE ALL BITS SET
3847 015144  001401                            BEQ     5$                      ;YES GO ON
3848                                                                    ;NO GO TO ERROR
3849 015146  104001                            ERROR   .1              ;CPU ERROR
3850 015150  005006                    5$:     CLR     R6                      ;SET USER STACK TO ALL ZEROES
3851 015152  022706  000000                    CMP     #0,R6                   ;ARE ALL BITS CLEARED
3852 015156  001401                            BEQ     6$                      ;YES GO ON
3853                                                                    ;NO GO TO ERROR
3854 015160  104001                            ERROR   .1              ;CPU ERROR
3855 015162  012706  125252            6$:     MOV     #125252,R6              ;SET USER STACK TO ALTERNATING PATTERN
3856 015166  022706  125252                    CMP     #125252,R6              ;IS USER SP CORRECT
3857 015172  001401                            BEQ     7$                      ;YES GO ON
3858                                                                    ;NO GO TO ERROR
3859 015174  104001                            ERROR   .1              ;CPU ERROR
3860 015176  012706  052525            7$:     MOV     #52525,R6               ;SET USER STACK TO ALTERNATING PATTERN
3861 015202  022706  052525                    CMP     #52525,R6               ;IS USER SP CORRECT
3862 015206  001401                            BEQ     8$                      ;YES GO ON
3863                                                                    ;NO GO TO ERROR
3864 015210  104001                            ERROR   .1              ;CPU ERROR
3865 015212  012706  000600            8$:     MOV     #600,R6                 ;SETUP USER SP
3866 015216  005037  177776                    CLR     @#177776                ;SET PS TO KER MODE
3867 015222  012706  001000                    MOV     #STBOT,R6               ;SETUP KERNEL SP
3868 015226  005037  000700                    CLR     @#700                   ;CLEAR FIRST WORDS OF SUP, KER, AND USE STACKS
3869 015232  005037  000600                    CLR     @#600                   ;
```

M6

********** DOUBLE OPERAND TESTS **********

```
3870 015236  005037  001000                    CLR     @#STBOT                 ;
3871 015242  004767  000054                    JSR     PC,CHECK                ; TEST KER, SUP, AND USE STACKS
3872 015246  012737  040000  177776  RET1:     MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3873 015254  022706  000700                    CMP     #700,R6                 ;IS SUP SP CORRECT
3874 015260  001401                            BEQ     1$                      ;YES GO ON
3875                                                                           ;NO GO TO ERROR
3876 015262  104001                            ERROR   .+1                     ;CPU ERROR
3877 015264  012737  140000  177776  1$:       MOV     #140000,@#177776        ;SET PSW TO USE MODE
3878 015272  022706  000600                    CMP     #600,R6                 ;IS USE SP CORRECT
3879 015276  001401                            BEQ     2$                      ;YES GO ON
3880                                                                           ;NO GO TO ERROR
3881 015300  104001                            ERROR   .+1                     ;CPU ERROR
3882 015302  005037  177776          2$:       CLR     @#177776                ;SET PSW TO KER MODE
3883 015306  022706  001000                    CMP     #STBOT,R6               ;IS KER SP CORRECT
3884 015312  001401                            BEQ     3$                      ;YES GO ON
3885                                                                           ;NO GO TO ERROR
3886 015314  104001                            ERROR   .+1                     ;CPU ERROR
3887 015316                          3$:
3888 015316  000167  000206                    JMP     MTS0
3889                                 ;
3890                                 ;ROUTINE TO CHECK STACKS AFTER TWO RTS STATEMENTS
3891                                 ;
3892 015322  012737  040000  177776  CHECK:    MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3893 015330  004767  000044                    JSR     PC,CHECK1               ; TEST SUP, KER, AND USE STACKS
3894 015334  022716  000000          RET2:     CMP     #0,(SP)                 ;IS SUP STACK CLEARED
3895 015340  001401                            BEQ     1$                      ;YES GO ON
3896                                                                           ;NO GO TO ERROR
3897 015342  104001                            ERROR   .+1                     ;CPU ERROR
3898 015344  012737  140000  177776  1$:       MOV     #140000,@#177776        ;SET PSW TO USE MODE
3899 015352  022716  000000                    CMP     #0,(SP)                 ;IS USE STACK CLEARED
3900 015356  001401                            BEQ     2$                      ;YES GO ON
3901                                                                           ;NO GO TO ERROR
3902 015360  104001                            ERROR   .+1                     ;CPU ERROR
3903 015362  005037  177776          2$:       CLR     @#177776                ;SET PSW TO KER MODE
3904 015366  022716  015246                    CMP     #RET1,(SP)              ;DOES KER STACK HAVE CORRECT DATA
3905 015372  001401                            BEQ     3$                      ;YES GO ON
3906                                                                           ;NO GO TO ERROR
3907 015374  104001                            ERROR   .+1                     ;CPU ERROR
3908 015376  000207                  3$:       RTS     PC                      ;RETURN
3909                                 ;
3910                                 ;ROUTINE TO CHECK STACKS AFTER ONE RTS
3911                                 ;
3912 015400  012737  140000  177776  CHECK1:   MOV     #140000,@#177776        ;SET PSW TO USE MODE
3913 015406  004767  000044                    JSR     PC,CHECK2               ; TEST KER, SUP, AND USE STACKS
3914 015412  022716  000000          RET3:     CMP     #0,(SP)                 ;IS USE STACK CLEARED
3915 015416  001401                            BEQ     1$                      ;YES GO ON
3916                                                                           ;NO GO TO ERROR
3917 015420  104001                            ERROR   .+1                     ;CPU ERROR
3918 015422  005037  177776          1$:       CLR     @#177776                ;SET PSW TO KER MODE
3919 015426  022716  015246                    CMP     #RET1,(SP)              ;IS KER STACK CORRECT
3920 015432  001401                            BEQ     2$                      ;YES GO ON
3921                                                                           ;NO GO TO ERROR
3922 015434  104001                            ERROR   .+1                     ;CPU ERROR
3923 015436  012737  040000  177776  2$:       MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3924 015444  022716  015334                    CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
3925 015450  001401                            BEQ     3$                      ;YES GO ON
3926                                                                           ;NO GO TO ERROR
```

N6


```
015452  104001                          ERROR   +1              ;CPU ERROR
015454  000207                  3$:     RTS     PC                      ;RETURN
                                ;
                                ;ROUTINE TO CHECK STACKS AFTER ZERO RTS
                                ;
015456  022716  015412          CHECK2: CMP     #RET3,(SP)              ;IS USE STACK CORRECT
015462  001401                          BEQ     1$                      ;YES GO ON
                                                                        ;NO GO TO ERROR
015464  104001                          ERROR   +1              ;CPU ERROR
015466  012737  040000  177776  1$:     MOV     #40000,@#177776         ;SET PSW TO SUP MODE
015474  022716  015334                  CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
015500  001401                          BEQ     2$                      ;YES GO ON
                                                                        ;NO GO TO ERROR
015502  104001                          ERROR   +1              ;CPU ERROR
015504  005037  177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
015510  022716  015246                  CMP     #RET1,(SP)              ;IS KER STACK CORRECT
015514  001401                          BEQ     3$                      ;YES GO ON
                                                                        ;NO GO TO ERROR
015516  104001                          ERROR   +1              ;CPU ERROR
015520  012737  140000  177776  3$:     MOV     #140000,@#177776        ;SET PSW TO USE MODE
015526  000207                          RTS     PC                      ;RETURN
                                ;
015530                          MTSO:
015530                          MMVCC:

                                ;       TEST MOV CONDITION CODES - **0-
015530  000277                          SCC                             ;
015532  000244                          CLZ                             ;CC=1011
015534  012704  000000                  MOV     #0,R4                   ;CC=0101, R4=0
015540  100403                          BMI     1$                      ;ERROR IF N FLAG
015542  102402                          BVS     1$                      ;ERROR IF V FLAG SET
015544  103001                          BCC     1$                      ;ERROR IF C FLAG CLEAR
015546  001401                          BEQ     2$                      ;SKIP IF Z FLAG SET
                                                                        ;CC SHOULD=0101
015550  104001                  1$:     ERROR   +1              ;CPU ERROR
015552  000277                  2$:     SCC
015554  000251                          .WORD   251                     ;CC=0110
015556  012704  100000                  MOV     #100000,R4              ;R4=100000, CC=1000
015562  001403                          BEQ     3$                      ;ERROR IF Z SET
015564  102402                          BVS     3$                      ;ERROR IF V SET
015566  103401                          BCS     3$                      ;ERROR IF C SET
015570  100401                          BMI     4$                      ;EXIT IF N SET
015572  104001                  3$:     ERROR   +1              ;CPU ERROR
                                                                ;CC SHOULD= 1000
015574                          4$:


015574                          MBTCC:

                                ;       TEST BIT CONDITION CODES - **0-
015574  005004                          CLR     R4
015576  005104                          COM     R4                      ;R4=-1
015600  000277                          SCC                             ;
015602  000244                          CLZ                             ;CC=1011
015604  032704  000000                  BIT     #0,R4                   ;CC=0101
015610  100403                          BMI     1$                      ;ERROR IF N FLAG
015612  102402                          BVS     1$                      ;ERROR IF V FLAG SET
```

```
015614  103001                          BCC     1$                      ;ERROR IF C FLAG CLEAR
015616  001401                          BEQ     2$                      ;SKIP IF Z FLAG SET
015620  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;CC SHOULD=0101
015622  000277                  2$:     SCC
015624  000251                          .WORD   251                     ;CC=0110
015626  032704  100000                  BIT     #100000,R4              ;CC=1000
015632  001403                          BEQ     3$                      ;ERROR IF Z SET
015634  102402                          BVS     3$                      ;ERROR IF V SET
015636  103401                          BCS     3$                      ;ERROR IF C SET
015640  100401                          BMI     4$                      ;EXIT IF N SET
015642  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;CC SHOULD= 1000
015644                          4$:


015644                          MBCCC:

                                ;       TEST BIC CONDITION CODES - **0-
015644  005004                          CLR     R4
015646  005104                          COM     R4                      ;R4=-1
015650  000277                          SCC                             ;
015652  000244                          CLZ                             ;CC=1011
015654  042704  177777                  BIC     #177777,R4              ;CC=0101
015660  100403                          BMI     1$                      ;ERROR IF N FLAG
015662  102402                          BVS     1$                      ;ERROR IF V FLAG SET
015664  103001                          BCC     1$                      ;ERROR IF C FLAG CLEAR
015666  001401                          BEQ     2$                      ;SKIP IF Z FLAG SET
015670  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;CC SHOULD=0101
015672  005104                  2$:     COM     R4                      ;R4=-1
015674  000277                          SCC
015676  000251                          .WORD   251                     ;CC=0110
015700  042704  077777                  BIC     #77777,R4               ;CC=1000
015704  001403                          BEQ     3$                      ;ERROR IF Z SET
015706  102402                          BVS     3$                      ;ERROR IF V SET
015710  103401                          BCS     3$                      ;ERROR IF C SET
015712  100401                          BMI     4$                      ;EXIT IF N SET
015714  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;CC SHOULD= 1000
015716                          4$:


015716                          MBSCC:

                                ;       TEST BIS CONDITION CODES
015716  005004                          CLR     R4                      ;R4=0
015720  000277                          SCC
015722  000246                          .WORD   246                     ;CC=1001
015724  052704  000000                  BIS     #0,R4                   ;R4=0, CC=0101
015730  100403                          BMI     1$                      ;ERROR IF MINUS
015732  102402                          BVS     1$                      ;ERROR IF V SET
015734  103001                          BCC     1$                      ;ERROR IF C CLEAR
015736  001401                          BEQ     2$                      ;BRANCH IF GOOD
015740  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BIS CC FAILED
015742  000277                  2$:     SCC
```

```
********** DOUBLE OPERAND TESTS **********

4049 015744  000241                    CLC                             ;CC=1110
4050 015746  052704  100076             BIS     #100076,R4              ;R4=100076, CC=1000
4051 015752  001403                     BEQ     3$                      ;ERROR IF Z SET
4052 015754  102402                     BVS     3$                      ;ERROR IF V SET
4053 015756  103401                     BCS     3$                      ;ERROR IF C SET
4054 015760  100401                     BMI     4$                      ;BRANCH IF GOOD
4055 015762  104001             3$:     ERROR   .1              ;CPU ERROR
4056                                                             ;BAD BIS CC
4057 015764                     4$:
4058
4059
4062 015764                     MDCCC:
4063
4064                            ;       TEST DEC, INC CONDITION CODES
4065 015764  012704  077777             MOV     #77777,R4               ;R4=77777
4066 015770  000257                     CCC
4067 015772  000261                     SEC                             ;CC=0001
4068 015774  005204                     INC     R4                      ;R4=100000, CC=0011
4069 015776  001403                     BEQ     1$                      ;ERROR IF ZERO
4070 016000  100002                     BPL     1$                      ;ERROR IF POSITIVE
4071 016002  102001                     BVC     1$                      ;ERROR IF V CLEAR
4072 016004  103401                     BCS     2$                      ;BRANCH IF GOOD
4073 016006  104001             1$:     ERROR   .1              ;CPU ERROR
4074                                                             ;INC FAILED
4075 016010  000257             2$:     CCC
4076 016012  005204                     INC     R4                      ;R4=100001, CC=1000
4077 016014  103413                     BCS     3$                      ;ERROR IF C SET
4078 016016  102412                     BVS     3$                      ;ERROR IF V SET
4079 016020  005304                     DEC     R4                      ;R4=100000, CC=1000
4080 016022  102410                     BVS     3$                      ;ERROR IF V SET
4081 016024  103407                     BCS     3$                      ;ERROR IF C SET
4082 016026  000277                     SCC                             ;
4083 016030  000252                     .WORD   252                     ;CC=0101
4084 016032  005304                     DEC     R4                      ;R4=77777, CC=1011
4085 016034  001403                     BEQ     3$                      ;ERROR IF Z SET
4086 016036  102002                     BVC     3$                      ;ERROR IF V CLEAR
4087 016040  103001                     BCC     3$                      ;ERROR IF C CLEAR
4088 016042  100001                     BPL     4$                      ;BRANCH IF GOOD
4089 016044  104001             3$:     ERROR   .1              ;CPU ERROR
4090                                                             ;BAD CC
4091 016046                     4$:
4092
4093
4103 016046                     MCTSCC:
4104
4105                            ;       TEST CLR, TST, SWAB CONDITION CODES
4106                            ;;**************************************************************
                                ;0100 - **00 - **00
4107 016046  000277                     SCC
4108 016050  000244                     CLZ                             ;CC=1011
4109 016052  005004                     CLR     R4                      ;R4=0, CC=0100
4110 016054  100403                     BMI     1$                      ;ERROR IF MINUS
4111 016056  102402                     BVS     1$                              ;ERROR IF V SET
4112 016060  103401                     BCS     1$                      ;ERROR IF C SET
4113 016062  001401                     BEQ     2$                      ;BRANCH IF GOOD
4114 016064  104001             1$:     ERROR   .1              ;CPU ERROR
4115                                                             ;BAD CC ON CLR
```

```
********** DOUBLE OPERAND TESTS **********

4116 016066  005104                  2$:     COM     R4                      ;R4=-1
4117 016070  000277                          SCC                             ;CC=1111
4118 016072  005704                          TST     R4                      ;CC=1000
4119 016074  001403                          BEQ     3$                      ;ERROR IF Z SET
4120 016076  102402                          BVS     3$                      ;ERROR IF V SET
4121 016100  103401                          BCS     3$                      ;ERROR IF C SET
4122 016102  100401                          BMI     4$                      ;BRANCH IF GOOD
4123 016104  104001                  3$:     ERROR   .1              ;CPU ERROR
4124                                                                 ;BAD TST CC
4125 016106  000277                  4$:     SCC                             ;CC=1111
4126 016110  000304                          SWAB    R4                      ;CC=1000
4127 016112  102402                          BVS     5$                      ;ERROR IF V SET
4128 016114  103401                          BCS     5$                      ;ERROR IF C SET
4129 016116  100401                          BMI     6$                      ;BRANCH IF GOOD
4130 016120  104001                  5$:     ERROR   .1              ;CPU ERROR
4131                                                                 ;BAD SWAB CC
4132 016122                          6$:
4133
4134
4137 016122                          MADCC:
4138
4139                                 ;       TEST ADD CONDITION CODES
4140 016122  012704  077777                  MOV     #77777,R4               ;R4=77777
4141 016126  012701  000001                  MOV     #1,R1                   ;R1=1
4142 016132  000257                          CCC                             ;CC=0000
4143 016134  060401                          ADD     R4,R1                   ;77777 + 1 = 100000 IN R1
4144 016136  102003                          BVC     1$                      ;ERROR IF V CLEAR
4145 016140  103402                          BCS     1$                      ;ERROR IF CARRY
4146 016142  001401                          BEQ     1$                      ;ERROR IF Z SET
4147 016144  100401                          BMI     2$                      ;BRANCH IF GOOD
4148 016146  104001                  1$:     ERROR   .1              ;CPU ERROR
4149                                                                 ;CC SHOULD =1010
4150 016150  005204                  2$:     INC     R4                      ;R4=100000
4151 016152  060401                          ADD     R4,R1                   ;100000 + 100000 = 0 IN R1
4152 016154  102002                          BVC     3$                      ;ERROR IF V CLEAR
4153 016156  103001                          BCC     3$                      ;ERROR IF CARRY  CLEAR
4154 016160  001401                          BEQ     4$                      ;BRANCH IF GOOD
4155 016162  104001                  3$:     ERROR   .1              ;CPU ERROR
4156                                                                 ;CC SHOULD = 0111
4157 016164  060401                  4$:     ADD     R4,R1                   ;0 + 100000 = 100000
4158 016166  102402                          BVS     5$                      ;ERROR IF V SET
4159 016170  103401                          BCS     5$                      ;ERROR IF  SET
4160 016172  100401                          BMI     6$                      ;BRANCH IF GOOD
4161 016174  104001                  5$:     ERROR   .1              ;CPU ERROR
4162                                                                 ;CC SHOULD = 1000
4163 016176                          6$:
4164
4165
4168 016176                          MACCC:
4169
4170                                 ;       TEST ADC CONDITION CODES
4171 016176  012704  177777                  MOV     #177777,R4              ;R4=177777
4172 016202  000277                          SCC                             ;CC=1111
4173 016204  005504                          ADC     R4                      ;R4=0 CC=0101
4174 016206  100403                          BMI     1$                      ;ERROR IF MINUS
4175 016210  102402                          BVS     1$                      ;ERROR IF V SET
4176 016212  103001                          BCC     1$                      ;ERROR IF C SET
```

```
016214  001401                          BEQ     2$                      ;BRANCH IF GOOD
016216  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ADC
016220  012704  077777          2$:     MOV     #077777,R4              ;R4=077777
016224  000277                          SCC                             ;
016226  000242                          CLV                             ;CC=1101
016230  005504                          ADC     R4                      ;R4=100000 CC=1010
016232  100003                          BPL     3$                      ;ERROR IF PLUS
016234  103402                          BCS     3$                      ;ERROR IF C SET
016236  001401                          BEQ     3$                      ;ERROR IF ZERO
016240  102401                          BVS     4$                      ;BRANCH IF GOOD
016242  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ADC
016244  000277                  4$:     SCC                             ;CC=1111
016246  005504                          ADC     R4                      ;R4=100000 CC=1000
016250  102402                          BVS     5$                      ;ERROR IF V SET
016252  103401                          BCS     5$                      ;ERROR IF C SET
016254  100401                          BMI     6$                      ;BRANCH IF GOOD
016256  104001                  5$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ADC CC SHOULD= 1000
016260                          6$:


016260                          MNCCCC:

                                ;       TEST NEG, CMP, COM CONDITION CODES
016260  012704  077777                  MOV     #077777,R4              ;R4=77777
016264  000257                          CCC                             ;CC=0000
016266  005404                          NEG     R4                      ;R4=100001      CC=1001
016270  102403                          BVS     1$                      ;ERROR IF V SET
016272  103002                          BCC     1$                      ;ERROR IF C CLEAR
016274  001401                          BEQ     1$                      ;ERROR IF Z SET
016276  100401                          BMI     2$                      ;BRANCH IF GOOD
016300  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD NEGATE
016302  005004                  2$:     CLR     R4                      ;R4=0
016304  000257                          CCC                             ;CC=0000
016306  005404                          NEG     R4                      ;CC=0101
016310  100403                          BMI     3$                      ;ERROR IF N SET
016312  103402                          BCS     3$                      ;ERROR IF C SET
016314  102401                          BVS     3$                      ;ERROR IF V SET
016316  001401                          BEQ     4$                      ;BRANCH IF GOOD
016320  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD NEG
016322  012704  077777          4$:     MOV     #77777,R4               ;R4=77777
016326  012701  170000                  MOV     #170000,R1              ;R1=170000
016332  000257                          CCC                             ;CC=0000
016334  020401                          CMP     R4,R1                   ;77777 - 170000 = 107777 CC= 1011
016336  102003                          BVC     5$                      ;ERROR IF V CLEAR
016340  103002                          BCC     5$                      ;ERROR IF C CLEAR
016342  001401                          BEQ     5$                      ;ERROR IF ZERO
016344  100401                          BMI     6$                      ;BRANCH IF GOOD
016346  104001                  5$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD CMP
016350  000257                  6$:     CCC
016352  005101                          COM     R1                      ;R1=7777
016354  100403                          BMI     7$                      ;ERROR IF MINUS
```

```
********** DOUBLE OPERAND TESTS ******************

4236 016356  001402                        BEQ     7$                      ;ERROR IF ZERO
4237 016360  103001                        BCC     7$                      ;ERROR IF CARRY
4238 016362  102001                        BVC     8$                      ;BRANCH IF GOOD
4239 016364  104001              7$:       ERROR   +1              ;CPU ERROR
4240                                                               ;BAD COM
4241 016366  000277              8$:       SCC
4242 016370  005101                        COM     R1
4243 016372  100401                        BMI     10$                     ;BRANCH IF GOOD
4244 016374  104001              9$:       ERROR   +1              ;CPU ERROR
4245                                                               ;BAD COM
4246 016376                      10$:
4247
4248
4251 016376                      MSBCC:
4252
4253                             ;         TEST SUB CONDITION CODES
4254 016376  012704  077775                MOV     #77775,R4               ;R4=77775
4255 016402  000257                        CCC                             ;CC=0000
4256 016404  162704  137757                SUB     #137757,R4              ;77775 - 137757
4257                                                                       ;TRY TO CAUSE AN ARITHMETIC OVERFLOW
4258 016410  102003                        BVC     1$                      ;ERROR IF V CLEAR
4259 016412  100002                        BPL     1$                      ;ERROR IF RESULT IS POSITIVE
4260 016414  001401                        BEQ     1$                      ;ERROR IF  Z SET
4261 016416  103401                        BCS     2$                      ;BRANCH IF GOOD
4262 016420  104001              1$:       ERROR   +1              ;CPU ERROR
4263                                                               ;BAD SUBTRACT
4264 016422  012704  000005      2$:       MOV     #5,R4                   ;R4=5
4265 016426  000257                        CCC                             ;CC=0000
4266 016430  162704  000012                SUB     #12,R4                  ;5-12=-5 AND SETS CARRY
4267 016434  103003                        BCC     3$                      ;ERROR IF CARRY CLEAR
4268 016436  102402                        BVS     3$                      ;ERROR IF OVERFLOW
4269 016440  001401                        BEQ     3$                      ;ERROR IF ZERO
4270 016442  100401                        BMI     4$                      ;BRANCH IF GOOD
4271 016444  104001              3$:       ERROR   +1              ;CPU ERROR
4272                                                               ;SUBTRACT FAILED
4273 016446                      4$:
4274
4275
4278 016446                      MSBCCC:
4279
4280                             ;         TEST SBC CONDITION CODES
4281 016446  012704  100000                MOV     #100000,R4              ;R4=100000
4282 016452  000257                        CCC                             ;C=0000
4283 016454  005604                        SBC     R4                      ;TRY TO SET V
4284 016456  100006                        BPL     1$                      ;ERROR IF N CLEAR
4285 016460  102405                        BVS     1$                      ;ERROR IF V SET (HAVENT SET C YET)
4286 016462  000261                        SEC                             ;CC SHOULD = 1001
4287 016464  005604                        SBC     R4                      ;TRY AGAIN TO SET V
4288 016466  102002                        BVC     1$                      ;ERROR IF V CLEAR
4289 016470  103401                        BCS     1$                      ;ERROR IF C SET
4290 016472  100001                        BPL     2$                      ;BRANCH IF GOOD
4291 016474  104001              1$:       ERROR   +1              ;CPU ERROR
4292                                                               ;SBC FAILED
4293 016476  005004              2$:       CLR     R4                      ;R4=0
4294 016500  000277                        SCC
4295 016502  000241                        CLC                             ;CC=1110
4296 016504  005604                        SBC     R4                      ;TRY TO CAUSE C FLAG FAILURE
```

```
016506  103410                          BCS     3$                      ;ERROR IF C SET
016510  102407                          BVS     3$                      ;ERROR IF V SET
016512  001006                          BNE     3$                      ;ERROR IF  NOT ZERO
016514  000261                          SEC                             ;SET CARRY
016516  005604                          SBC     R4                      ;NOW, 0 - CARRY = 177777
016520  103003                          BCC     3$                      ;ERROR IF CARRY CLEAR
016522  102402                          BVS     3$                      ;ERROR IF V SET
016524  001401                          BEQ     3$                      ;ERROR IF ZERO
016526  100401                          BMI     4$                      ;BRANCH IF GOOD
016530  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;SBC FAILED
016532                          4$:


016532                          MRLCC:

                                ;       TEST ROL CONDITION CODES
016532  012704  060000                  MOV     #60000,R4               ;R4= 0110000000000000
016536  000257                          CCC                             ;CC=0000
016540  006104                          ROL     R4                      ;R4= 1100000000000000
016542  103402                          BCS     1$                      ;ERROR IF CARRY
016544  102001                          BVC     1$                      ;ERROR IF V CLEAR
016546  100401                          BMI     2$                      ;BRANCH IF GOOD
016550  104001                  1$:     ERROR   +1                  ;CPU ERROR
                                                                    ;ROL  FAILED
016552  006104                  2$:     ROL     R4                      ;R4= 1000000000000000
016554  103002                          BCC     3$                      ;ERROR IF CARRY CLEAR
016556  102401                          BVS     3$                      ;ERROR IF V SET
016560  100401                          BMI     4$                      ;BRANCH IF GOOD
016562  104001                  3$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ROL
016564  006104                  4$:     ROL     R4                      ;R4 = 0000000000000001
016566  102003                          BVC     5$                      ;ERROR IF V CLEAR
016570  103002                          BCC     5$                      ;ERROR IF C CLEAR
016572  100401                          BMI     5$                      ;ERROR IF MINUS
016574  001001                          BNE     6$                      ;BRANCH IF GOOD
016576  104001                  5$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ROL
016600  006104                  6$:     ROL     R4                      ;R4=000000000000011
016602  102402                          BVS     7$                      ;ERROR IF V SET
016604  103401                          BCS     7$                      ;ERROR IF C SET
016606  100001                          BPL     8$                      ;BRANCH IF GOOD
016610  104001                  7$:     ERROR   +1                  ;CPU ERROR
                                                                    ;BAD ROL
016612                          8$:


016612                          MRRCC:

                                ;       TEST ROR CONDITION CODES
016612  012704  000003                  MOV     #3,R4                   ;R4= 0000000000000011
016616  000257                          CCC                             ;CC= 0000
016620  006004                          ROR     R4                      ;R4= 0000000000000001
016622  103002                          BCC     1$                      ;ERROR IF NO CARRY
016624  102001                          BVC     1$                      ;ERROR IF V CLEAR
016626  100001                          BPL     2$                      ;BRANCH IF GOOD
016630  104001                  1$:     ERROR   +1                  ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS ******************

4358
4359 016632  006004                2$:     ROR     R4              ;ROR  FAILED
4360 016634  103002                        BCC     3$              ;R4= 1000000000000000
4361 016636  102401                        BVS     3$              ;ERROR IF CARRY CLEAR
4362 016640  100401                        BMI     4$              ;ERROR IF V SET
4363 016642  104001                3$:     ERROR   ;1              ;BRANCH IF GOOD
4364                                                               ;CPU ERROR
4365 016644  006004                4$:     ROR     R4              ;BAD ROR
4366 016646  102002                        BVC     5$              ;R4 = 1100000000000000
4367 016650  103401                        BCS     5$              ;ERROR IF V
4368 016652  100401                        BMI     6$              ;ERROR IF C SET
4369 016654  104001                5$:     ERROR   ;1              ;BRANCH IF GOOD
4370                                                               ;CPU ERROR
4371 016656  006004                6$:     ROR     R4              ;BAD ROR
4372 016660  102402                        BVS     7$              ;R4= 0110000000000000
4373 016662  103401                        BCS     7$              ;ERROR IF V SET
4374 016664  100001                        BPL     8$              ;ERROR IF C SET
4375 016666  104001                7$:     ERROR   ;1              ;BRANCH IF GOOD
4376                                                               ;CPU ERROR
4377 016670                        8$:                             ;BAD ROR
4378
4379
4382 016670                        XCBIT:

                                   ;       TEST C BIT WITH ROR/ROL
                                   ;;**************************************************************
                                   ;THIS TEST IS TO CHECK FOR A SLOW C BIT PATH INTERNAL TO THE J11 DATA CHIP
                                   ;PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 AND 1593)
4391
4392
4393
4394 016670  012701  052525                MOV     #52525,R1       ;INIT R1 WITH DATA
4395 016674  000241                        CLC                     ;CLEAR THE C BIT
4396 016676  006001                        ROR     R1              ;R1=025252, C BIT =1
4397 016700  006001                        ROR     R1              ;R1=112525, C BIT =0
4398 016702  006001                        ROR     R1              ;R1=045252, C BIT =1
4399 016704  103401                        BCS     1$              ;BRANCH IF CARRY BIT SET
4400 016706  104001                        ERROR   ;1              ;CPU ERROR
4401 016710  022701  045252        1$:     CMP     #45252,R1       ;IS DATA IN R1 = TO EXPECTED DATA?
4402 016714  001401                        BEQ     2$              ;BRANCH IF YES
4403 016716  104001                        ERROR   ;1              ;CPU ERROR
4404 016720  012701  125252        2$:     MOV     #125252,R1      ;SET UP R1
4405 016724  000241                        CLC                     ;CLEAR THE CARRY BIT
4406 016726  006101                        ROL     R1              ;R1=052524, C BIT =1
4407 016730  006101                        ROL     R1              ;R1=125251, C BIT =0
4408 016732  006101                        ROL     R1              ;R1=052522, C BIT =1
4409 016734  103401                        BCS     3$              ;BRANCH IF CARRY SET
4410 016736  104001                        ERROR   ;1              ;CPU ERROR
4411 016740  022701  052522        3$:     CMP     #052522, R1     ;IS DATA IN R1 = TO EXPECTED DATA?
4412 016744  001401                        BEQ     4$
4413 016746  104001                        ERROR   ;1              ;CPU ERROR
4414 016750                        4$:
4416
4418 016750                        MALCC:
4419
4420                               ;       TEST ASL CONDITION CODES
4421 016750  012704  060000                MOV     #60000,R4       ;R4= 0110000000000000
4422 016754  000257                        CCC                     ;CC=0000
4423 016756  006304                        ASL     R4              ;C=0 R4= 1100000000000000
4424 016760  103402                        BCS     1$              ;ERROR IF CARRY
```

```
********** DOUBLE OPERAND TESTS ******************

4425 016762  102001                    BVC     1$              ;ERROR IF V CLEAR
4426 016764  100401                    BMI     2$              ;BRANCH IF GOOD
4427 016766  104001            1$:     ERROR   +1          ;CPU ERROR
4428                                                       ;ASL  FAILED
4429 016770  006304            2$:     ASL     R4              ;C=1 R4= 1000000000000000
4430 016772  103002                    BCC     3$              ;ERROR IF CARRY CLEAR
4431 016774  102401                    BVS     3$              ;ERROR IF V SET
4432 016776  100401                    BMI     4$              ;BRANCH IF GOOD
4433 017000  104001            3$:     ERROR   +1          ;CPU ERROR
4434                                                       ;BAD ASL
4435 017002  006304            4$:     ASL     R4              ;C=1 R4= 0000000000000000
4436 017004  102003                    BVC     5$              ;ERROR IF V CLEAR
4437 017006  103002                    BCC     5$              ;ERROR IF C CLEAR
4438 017010  100401                    BMI     5$              ;ERROR IF MINUS
4439 017012  001401                    BEQ     6$              ;BRANCH IF GOOD
4440 017014  104001            5$:     ERROR   +1          ;CPU ERROR
4441                                                       ;BAD ASL
4442 017016  006304            6$:     ASL     R4              ;C=0 R4= 0000000000000000
4443 017020  102402                    BVS     7$              ;ERROR IF V SET
4444 017022  103401                    BCS     7$              ;ERROR IF C SET
4445 017024  100001                    BPL     8$              ;BRANCH IF GOOD
4446 017026  104001            7$:     ERROR   +1          ;CPU ERROR
4447                                                       ;BAD ASL
4448 017030                    8$:
4449
4452 017030                    MARCC:
4453
4454                           ;       TEST ASR CONDITION CODES
4455 017030  012704  000341            MOV     #341,R4         ;R4= 0000000011100001
4456 017034  000257                    CCC                     ;CC=0000
4457 017036  006204                    ASR     R4              ;R4= 0000000001110000
4458 017040  103002                    BCC     1$              ;ERROR IF NO CARRY
4459 017042  102001                    BVC     1$              ;ERROR IF V CLEAR
4460 017044  100001                    BPL     2$              ;BRANCH IF GOOD
4461 017046  104001            1$:     ERROR   +1          ;CPU ERROR
4462                                                       ;ASR  FAILED
4463 017050  052704  100001    2$:     BIS     #100001,R4      ;R4= 1000000001110001
4464 017054  006204                    ASR     R4              ;R4= 1100000000111000
4465 017056  103002                    BCC     3$              ;ERROR IF CARRY CLEAR
4466 017060  102401                    BVS     3$              ;ERROR IF V SET
4467 017062  100401                    BMI     4$              ;BRANCH IF GOOD
4468 017064  104001            3$:     ERROR   +1          ;CPU ERROR
4469                                                       ;BAD ASR
4470 017066  006204            4$:     ASR     R4              ;R4= 1110000000011100
4471 017070  102002                    BVC     5$              ;ERROR IF V
4472 017072  103401                    BCS     5$              ;ERROR IF C SET
4473 017074  100401                    BMI     6$              ;BRANCH IF GOOD
4474 017076  104001            5$:     ERROR   +1          ;CPU ERROR
4475                                                       ;BAD ASR
4476 017100  006204            6$:     ASR     R4              ;R4= 1111000000001110
4477 017102  102005                    BVC     7$              ;ERROR IF V CLEAR
4478 017104  103404                    BCS     7$              ;ERROR IF C SET
4479 017106  100003                    BPL     7$              ;ERROR IF PLUS
4480 017110  022704  170016            CMP     #170016,R4      ;SEE IF EXPECTED RESULT
4481 017114  001401                    BEQ     8$              ;BRANCH IF GOOD
4482 017116  104001            7$:     ERROR   +1          ;CPU ERROR
4483                                                       ;BAD ASR
```

```
********** DOUBLE OPERAND TESTS ******************

4484 017120                          8$:
4485
4486
4489 017120                          MSXTCC:
4490
4491                                 ;       TEST SXT CONDITION CODES / -*0-
4492 017120  012704  123456                  MOV     #123456,R4                      ;R4=123456
4493 017124  010401                          MOV     R4,R1                   ;SAVE  CONTENTS
4494 017126  000257                          CCC                             ;CC=0000
4495 017130  006704                          SXT     R4                      ;R4=0 CC=0100
4496 017132  103403                          BCS     1$                      ;ERROR IF CARRY
4497 017134  100402                          BMI     1$                      ;ERROR IF MINUS
4498 017136  102401                          BVS     1$                      ;ERROR IF OVERFLOW
4499 017140  001401                          BEQ     2$                      ;BRANCH IF GOOD
4500 017142  104001                  1$:     ERROR   +1              ;CPU ERROR
4501                                                                 ;BAD SXT
4502 017144  010104                  2$:     MOV     R1,R4                   ;RESTORE R4
4503 017146  000277                          SCC                             ;CC=1111
4504 017150  006704                          SXT     R4                      ;R4=-1 CC=1001
4505 017152  001405                          BEQ     3$                      ;ERROR IF ZERO
4506 017154  100004                          BPL     3$                      ;ERROR IF PLUS
4507 017156  103003                          BCC     3$                      ;ERROR IF NO CARRY
4508 017160  102402                          BVS     3$                      ;ERROR IF OVERFLOW
4509 017162  005104                          COM     R4                      ;R4=0
4510 017164  001401                          BEQ     4$                      ;BRANCH IF GOOD
4511 017166  104001                  3$:     ERROR   +1              ;CPU ERROR
4512                                                                 ;BAD SXT
4513 017170                          4$:
4514
4515
4518 017170                          MXRCC:
4519
4520                                 ;       TEST XOR CONDITION CODES / **0-
4521 017170  012704  123456                  MOV     #123456,R4                      ;R4=123456
4522 017174  012701  052525                  MOV     #52525,R1                       ;R1=52525
4523 017200  000257                          CCC                             ;CC=0000
4524 017202  074104                          XOR     R1,R4                   ;*TI* R4=171173
4525 017204  102403                          BVS     1$                      ;;ERROR IF OVERFLOW
4526 017206  001402                          BEQ     1$                      ;ERROR IF ZERO
4527 017210  103401                          BCS     1$                      ;ERROR IF CARRY
4528 017212  100401                          BMI     2$                      ;BRANCH IF GOOD
4529 017214  104001                  1$:     ERROR   +1              ;CPU ERROR
4530                                                                 ;BAD XOR
4531 017216  012701  125252          2$:     MOV     #125252,R1                      ;R1=125252
4532 017222  000277                          SCC                             ;CC=1111
4533 017224  074104                          XOR     R1,R4                   ;R4=054321
4534 017226  100403                          BMI     3$                      ;ERROR IF MINUS
4535 017230  001402                          BEQ     3$                      ;ERROR IF ZERO
4536 017232  103001                          BCC     3$                      ;ERROR IF CARRY CLEAR
4537 017234  102001                          BVC     4$                      ;BRANCH IF GOOD
4538 017236  104001                  3$:     ERROR   +1              ;CPU ERROR
4539                                                                 ;BAD XOR
4540 017240  074404                  4$:     XOR     R4,R4                   ;R4=0
4541 017242  102406                          BVS     5$                      ;ERROR IF OVREFLOW
4542 017244  100405                          BMI     5$                      ;ERROR IF MINUS
4543 017246  103004                          BCC     5$                      ;ERROR IF NO CARRY
4544 017250  001003                          BNE     5$                      ;ERROR IF NOT ZERO
```

```
********** DOUBLE OPERAND TESTS ******************

4545 017252  022704  000000                CMP     #0,R4                   ;SEE IF EXPECTED RESULT
4546 017256  001401                        BEQ     6$                      ;BRANCH IF GOOD
4547 017260  104001              5$:       ERROR   +1              ;CPU ERROR
4548                                                               ;BAD XOR
4549 017262                      6$:
4550
4551
4552                             ;
4564 017262                      MSXT:
4565
4566                             ;       TEST SXT (SIGN EXTEND INSTRUCTION)
4567                             ;;**************************************************************
                                 ;AN ADDITIONAL TEST IS INCLUDED TO CHECK FOR A SLOW N BIT PATH
                                 ;ON A TRANSITION FROM ZERO TO ONE INTERNAL TO THE J11 DATA CHIP
                                 ;THE PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 OR 1593)
4568 017262  005004                        CLR     R4                      ;TRASH R4
4569 017264  000257                        CCC                             ;CC=0000
4570 017266  000271                        .WORD   271                     ;CC=1001
4571 017270  006704                        SXT     R4                      ;*TEST INSTRUCTION
4572 017272  102405                        BVS     1$                      ;BRANCH IF OVERFLOW IS NOT CLEARED
4573 017274  100004                        BPL     1$                      ;BRANCH IF N BIT EFFECTED
4574 017276  001403                        BEQ     1$                      ;BRANNCH IF Z BIT EFFECTED
4575 017300  103002                        BCC     1$                      ;BRANCH IF C BIT EFFECTED
4576 017302  005204                        INC     R4
4577 017304  001401                        BEQ     2$                      ;BRANCH IF R4 =0
4578 017306  104001              1$:       ERROR   +1                      ;CPU ERROR
4579                                                       ;CC SHOULD HAVE = 1101
4580 017310  000277              2$:       SCC
4581 017312  000250                        CLN                             ;CC=0111
4582 017314  005004                        CLR     R4
4583 017316  012714  000055                MOV     #55,(R4)                ;TRASH R4
4584 017322  006714                        SXT     (R4)                    ;*TEST INSTRUCTION
4585 017324  001005                        BNE     3$                      ;BRANCH IF BIT EFFECTED
4586 017326  102404                        BVS     3$                      ;BRANCH IF OVERFLOW
4587 017330  103403                        BCS     3$                      ;
4588 017332  100402                        BMI     3$                      ;BRANCH IF N IS SET
4589 017334  005714                        TST     (R4)                    ;VERIFY INSTRUCTION WORKED
4590 017336  001401                        BEQ     4$                      ;BRANCH IF R4=0
4591 017340  104001              3$:       ERROR   +1                      ;CPU ERROR
4592                                                       ;SXT FAILED
4593
4594                             ;       NOW TEST FOR SLOW N BIT IN J11 DATA CHIP
4595
4596 017342  012700  177777      4$:       MOV     #-1,R0                  ;R0=177777, N BIT = 1
4597 017346  005004                        CLR     R4                      ;CLEAT THE N BIT
4598 017350  006700                        SXT     R0                      ;****TEST INSTRUCTION***
4599                                                                       ;TEST N BIT TRANSITION 1 TO 0
4600 017352  005700                        TST     R0                      ;R0 SHOULD = 0
4601 017354  001401                        BEQ     5$                      ;BRANCH IF OK
4602 017356  104001                        ERROR   +1                      ;CPU ERROR
4603 017360  005000              5$:       CLR     R0                      ;CLEAR R0, N BIT = 0
4604 017362  012704  177777                MOV     #-1,R4                  ;SET N BIT
4605 017366  006700                        SXT     R0                      ;***TEST INSTRUCTION***
4606                                                                       ;TEST N BIT TRANSITION 0 TO 1
4607 017370  022700  177777                CMP     #-1,R0                  ;R0 SHOULD = 177777
4608 017374  001401                        BEQ     6$                      ;BRANCH IF OK
4609 017376  104001                        ERROR   +1                      ;CPU ERROR
```

L7


```
********** DOUBLE OPERAND TESTS ******************

4610 017400                          6$:
4612                                 ;
4614 017400                          MXOR:
4615
4616                                 ;       TEST XOR
4617 017400  012701  007643                     MOV     #7643,R1                ;SETUP DATA
4618 017404  012704  133333                     MOV     #133333,R4              ;SETUP DATA
4619 017410  000277                             SCC
4620 017412  074401                             XOR     R4,R1                   ;*TEST INSTRUCTION
4621 017414  100006                             BPL     1$                      ;BRANCH IF PLUS TO ERROR
4622 017416  001405                             BEQ     1$                      ;ERROR IF ZERO
4623 017420  103004                             BCC     1$                      ;ERROR IF CARRY CLEAR
4624 017422  102403                             BVS     1$                      ;ERROR IF V SET
4625 017424  020127  134570                     CMP     R1,#134570              ;VERIFY CORRECT RESULT
4626 017430  001401                             BEQ     2$                      ;BRANCH IF GOOD
4627 017432  104001                  1$:        ERROR   +1              ;CPU ERROR
4628                                                                    ;BAD XOR
4629 017434  010102                  2$:        MOV     R1,R2
4630 017436  000257                             CCC
4631 017440  074402                             XOR     R4,R2                   ;*TEST INSTRUCTION
4632 017442  100405                             BMI     3$                      ;ERROR IF MINUS
4633 017444  102404                             BVS     3$                      ;ERROR IF OVERFLOW
4634 017446  103403                             BCS     3$                      ;ERROR IF CARRY
4635 017450  020227  007643                     CMP     R2,#7643
4636 017454  001401                             BEQ     4$                      ;BRANCH IF GOOD
4637 017456  104001                  3$:        ERROR   +1              ;CPU ERROR
4638                                                                    ;BAD XOR
4639 017460                          4$:
4640
4642                                 ;
4644 017460                          MSOB:
4645
4646                                 ;       TEST SOB
4647 017460  012704  000555                     MOV     #555,R4                 ;SETUP TEST COUNTER
4648 017464  000277                             SCC                             ;CC=17
4649 017466  103015                  1$:        BCC     2$                      ;ERROR IF CARRY CLEAR
4650 017470  102014                             BVC     2$                      ;ERROR IF NO OVERFLOW
4651 017472  100013                             BPL     2$                      ;ERROR IF PLUS
4652 017474  001012                             BNE     2$                      ;ERROR IF ZERO
4653 017476  077405                             SOB     R4,1$                   ;*TEST INSTRUCTION
4654 017500  103005                             BCC     3$                      ;ERROR IF CARRY CLEAR
4655 017502  102004                             BVC     3$                      ;ERROR IF NO OVERFLOW
4656 017504  100003                             BPL     3$                      ;ERROR IF PLUS
4657 017506  001002                             BNE     3$                      ;ERROR IF ZERO
4658 017510  000167  000010                     JMP     4$
4659 017514  104001                  3$:        ERROR   +1                      ;CPU ERROR
4660                                                                    ;CC EFFECTED DURING TEST
4661 017516  000167  000002                     JMP     4$
4662 017522  104001                  2$:        ERROR   +1                      ;CPU ERROR
4663                                                                    ;CC EFFECTED AFTER TEST
4664 017524  020427  000000          4$:        CMP     R4,#0                   ;IS R4 CORRECT
4665 017530  001401                             BEQ     5$                      ;YES GO ON
4666 017532  104001                             ERROR   +1                      ;CPU ERROR
4667                                                                    ;NO GO TO ERROR
4668 017534                          5$:
4669
4671
```

********** DOUBLE OPERAND TESTS **********

```
4673 017534                            MMARK:
4674
4675                                   ;       MARK INSTRUCTION TEST
4676 017534  012706  000700                    MOV     #STBOT-100,SP           ;SETUP TEST STACK = 700
4677 017540  012737  125252  000776            MOV     #125252,@#STBOT-2       ;SET UP NEW R5 VALUE ON STACK
4678 017546  012705  017564                    MOV     #1$,R5                  ;PUT NEW PC IN R5
4679 017552  012746  006437                    MOV     #MARK+37,-(SP)          ;INSERT MARK 37 INSTRUCTION ONTO STACK
4680 017556  000277                            SCC
4681 017560  000116                            JMP     (SP)                    ;* TEST INSTRUCTION
4682                                                                           ;MARK INSTRUCTION SHOULD HAVE GONE TO 1$
4683 017562  104001                            ERROR   +1              ;CPU ERROR
4684                                   ;
4685 017564  101002                    1$:     BHI     2$                      ;ERROR IF C OR Z BIT CLEAR
4686 017566  100001                            BPL     2$                      ;ERROR IF N BIT CLEAR
4687 017570  102401                            BVS     3$                      ;BRANCH IF V BIT SET
4688                                                                           ;BAD CONDITION CODES ON MARK
4689 017572  104001                    2$:     ERROR   +1              ;CPU ERROR
4690 017574  022705  125252            3$:     CMP     #125252,R5              ;VERIFY R5
4691 017600  001401                            BEQ     4$                      ;BRANCH IF GOOD
4692 017602  104001                            ERROR   +1              ;CPU ERROR
4693
4694 017604  020627  001000            4$:     CMP     SP,#STBOT               ;VERIFY THAT STACK IS CORRECT
4695 017610  001401                            BEQ     15$                     ;BRANCH IF OK
4696                                                                           ;ERROR! STACK WAS NOT CORRECT AFTER MARK
4697 017612  104001                            ERROR   +1              ;CPU ERROR
4698
4699 017614  012746  052525            15$:    MOV     #52525,-(SP)            ;SETUP EXPECTED R5
4700 017620  012746  006400                    MOV     #6400,-(SP)             ;MOVE MARK 0 INSTRUCTION ON STACK
4701 017624  010605                            MOV     SP,R5                   ;R5=ADDRESS OF INSTRUCTION
4702 017626  004767  000004                    JSR     PC,5$                   ;LEAVE 6$ ON STACK
4703 017632  000167  000006            6$:     JMP     16$                     ;MARK RETURNED CORRECTLY
4704
4705 017636  000257                    5$:     CCC                             ;CLEAR THE CONDITION CODES
4706 017640  000205                            RTS     R5                      ;RETURN TO MARK INSTRUCTION
4707                                                                           ;NEXT INSTRUCTION ON STACK IS THE RETURN
4708                                                                           ;FROM THE JSR
4709                                                                           ;ERROR! BAD MARK SEQUENCE
4710 017642  104001                            ERROR   +1              ;CPU ERROR
4711
4712 017644  101402                    16$:    BLOS    7$                      ;IS C OR Z BIT SET?
4713 017646  100401                            BMI     7$                      ;IS N BIT SET?
4714 017650  102001                            BVC     8$                      ;IS V BIT SET?
4715                                                                           ;ERROR! CONDITIONS CODES INCORRECT
4716 017652  104001                    7$:     ERROR   +1              ;CPU ERROR
4717
4718 017654  020627  001000            8$:     CMP     R6,#STBOT               ;IS THE STACK CORRECT?
4719 017660  001401                            BEQ     9$                      ;BRANCH IF YES
4720                                                                           ;ERROR! BAD STACK CLEANUP
4721 017662  104001                            ERROR   +1              ;CPU ERROR
4722 017664  022705  052525            9$:     CMP     #52525,R5               ;VERIFY THAT R5 WAS LOADED PROPERLY.
4723 017670  001401                            BEQ     10$                     ;IF OK, GO TO NEXT TEST.
4724                                                                           ;ERROR! R5 WAS NOT CORRECT AFTER MARK.
4725 017672  104001                            ERROR   +1              ;CPU ERROR
4726 017674                            10$:
4728 017674                            XME100:
4730
4731                                   ;       TEST CLEAR CONDITION CODES INSTRUCTION
```

```
********** DOUBLE OPERAND TESTS ******************

4732 017674  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4733 017702  000257                          CCC                             ; TEST INSTRUCTION
4734 017704  022737  030000  177776          CMP     #30000,@#177776         ;DID IT CLEAR ALL CONDITION CODE BITS
4735 017712  001401                          BEQ     1$                      ;YES GO ON
4736 017714  104001                          ERROR   +1              ;CPU ERROR
4737                                                                 ;NO GO TO ERROR
4738 017716                          1$:
4739                                 ;
4741 017716                          XME101:
4743
4744                                 ;       TEST CLEAR C BIT INSTRUCTION
4745 017716  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW,
4746 017724  000241                          CLC                             ; TEST INSTRUCTION
4747 017726  022737  030016  177776          CMP     #30016,@#177776         ;DID IT CLEAR CARRY BIT
4748 017734  001401                          BEQ     1$                      ;YES GO ON
4749                                                                         ;C BIT NOT CLEAR GO TO ERROR
4750 017736  104001                          ERROR   +1              ;CPU ERROR
4751 017740                          1$:
4752                                 ;
4755 017740                          TE102:
4756                                 ;       TEST CLEAR N BIT INST (CLN)
4757 017740  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4758 017746  000250                          CLN                             ; TEST INSTRUCTION
4759 017750  022737  030007  177776          CMP     #30007,@#177776         ;DID IT CLEAR NEGATIVE BIT
4760 017756  001401                          BEQ     1$                      ;YES GO ON
4761 017760  104001                          ERROR   +1              ;CPU ERROR
4762                                                                 ;NO GO TO ERROR
4763 017762                          1$:
4764                                 ;
4767 017762                          TE103:
4768                                 ;       TEST CLEAR V BIT INST (CLV)
4769 017762  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4770 017770  000242                          CLV                             ; TEST INSTRUCTION
4771 017772  022737  030015  177776          CMP     #30015,@#177776         ;DID IT CLEAR OVERFLOW BIT
4772 020000  001401                          BEQ     1$                      ;YES GO ON
4773 020002  104001                          ERROR   +1              ;CPU ERROR
4774                                                                 ;NO GO TO ERROR
4775 020004                          1$:
4776                                 ;
4779 020004                          TE104:
4780                                 ;       TEST CLEAR Z BIT INST (CLZ)
4781 020004  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4782 020012  000244                          CLZ                             ; TEST INSTRUCTION
4783 020014  022737  030013  177776          CMP     #30013,@#177776         ;DID IT CLEAR ZERO BIT
4784 020022  001401                          BEQ     1$                      ;YES GO ON
4785 020024  104001                          ERROR   +1              ;CPU ERROR
4786                                                                 ;NO GO TO ERROR
4787 020026                          1$:
4788                                 ;
4791 020026                          TE105:
4792                                 ;       TEST SET CONDITION CODES INST (SCC)
4793 020026  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
4794 020034  000277                          SCC                             ; TEST INSTRUCTION
4795 020036  022737  030017  177776          CMP     #30017,@#177776         ;DID IT SET ALL CONDITION CODE BITS
4796 020044  001401                          BEQ     1$                      ;YES GO ON
4797 020046  104001                          ERROR   +1              ;CPU ERROR
4798                                                                 ;NO GO TO ERROR
```

```
********** DOUBLE OPERAND TESTS **********

4799 020050                                1$:
4800                                       ;
4803 020050                                TE106:
4804                                       ;       TEST SET C BIT INST (SEC)
4805 020050  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4806 020056  000261                                SEC                             ; TEST INSTRUCTION
4807 020060  022737  030001  177776                CMP     #30001,@#177776         ;DID IT SET THE CARRY BIT
4808 020066  001401                                BEQ     1$                      ;YES GO ON
4809 020070  104001                                ERROR   ;1              ;CPU ERROR
4810                                                                       ;NO GO TO ERROR
4811 020072                                1$:
4812                                       ;
4815 020072                                TE107:
4816                                       ;       TEST SET N BIT INST (SEN)
4817 020072  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4818 020100  000270                                SEN                             ; TEST INSTRUCTION
4819 020102  022737  030010  177776                CMP     #30010,@#177776         ;DID IT SET THE NEGATIVE BIT
4820 020110  001401                                BEQ     1$                      ;YES GO ON
4821 020112  104001                                ERROR   ;1              ;CPU ERROR
4822                                                                       ;NO GO TO ERROR
4823 020114                                1$:
4824                                       ;
4827 020114                                TE110:
4828                                       ;       TEST SET V BIT INST (SEV)
4829 020114  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4830 020122  000262                                SEV                             ; TEST INSTRUCTION
4831 020124  022737  030002  177776                CMP     #30002,@#177776         ;DID IT SET THE OVERFLOW BIT
4832 020132  001401                                BEQ     1$                      ;YES GO ON
4833 020134  104001                                ERROR   ;1              ;CPU ERROR
4834                                                                       ;NO GO TO ERROR
4835 020136                                1$:
4836                                       ;
4839 020136                                TE111:
4840                                       ;       TEST SET Z BIT INST (SEZ)
4841 020136  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4842 020144  000264                                SEZ                             ; TEST INSTRUCTION
4843 020146  022737  030004  177776                CMP     #30004,@#177776         ;DID IT SET THE ZERO BIT
4844 020154  001401                                BEQ     1$                      ;YES GO ON
4845 020156  104001                                ERROR   ;1              ;CPU ERROR
4846                                                                       ;NO GO TO ERROR
4847 020160                                1$:
4848                                       ;
4851 020160                                TE112:
4852                                       ;       TEST MULTIPLE CLEARS OF CC BITS
4853 020160  012737  030000  177776                MOV     #30000,@#177776         ;INIT PSW
4854 020166  000277                                SCC                             ;SETUP PSW
4855 020170  000243                                .WORD   243                     ; TEST CLC CLV
4856 020172  022737  030014  177776                CMP     #30014,@#177776         ;PSW CORRECT?
4857 020200  001401                                BEQ     1$                      ;YES GO ON
4858 020202  104001                                ERROR   ;1              ;CPU ERROR
4859                                                                       ;NO GO TO ERROR
4860 020204  000277                        1$:     SCC                             ;SETUP PSW
4861 020206  000245                                .WORD   245                     ; TEST CLC CLZ
4862 020210  022737  030012  177776                CMP     #30012,@#177776         ;PSW CORRECT?
4863 020216  001401                                BEQ     2$                      ;YES GO ON
4864 020220  104001                                ERROR   ;1              ;CPU ERROR
4865                                                                       ;NO GO TO ERROR
```

```
********** DOUBLE OPERAND TESTS **********

4866 020222  000277                    2$:     SCC                             ;SETUP PSW
4867 020224  000246                            .WORD   246                     ; TEST CLV CLZ
4868 020226  022737  030011  177776            CMP     #30011,@#177776         ;PSW CORRECT?
4869 020234  001401                            BEQ     3$                      ;YES GO ON
4870 020236  104001                            ERROR   .1              ;CPU ERROR
4871                                                                   ;NO GO TO ERROR
4872 020240  000277                    3$:     SCC                             ;SETUP PSW
4873 020242  000247                            .WORD   247                     ; TEST CLC CLV CLZ
4874 020244  022737  030010  177776            CMP     #30010,@#177776         ;PSW CORRECT?
4875 020252  001401                            BEQ     4$                      ;YES GO ON
4876 020254  104001                            ERROR   .1              ;CPU ERROR
4877                                                                   ;NO GO TO ERROR
4878 020256  000277                    4$:     SCC                             ;SETUP PSW
4879 020260  000251                            .WORD   251                     ; TEST CLN CLC
4880 020262  022737  030006  177776            CMP     #30006,@#177776         ;PSW CORRECT?
4881 020270  001401                            BEQ     5$                      ;YES GO ON
4882 020272  104001                            ERROR   .1              ;CPU ERROR
4883                                                                   ;NO GO TO ERROR
4884 020274  000277                    5$:     SCC                             ;SETUP PSW
4885 020276  000252                            .WORD   252                     ; TEST CLN CLV
4886 020300  022737  030005  177776            CMP     #30005,@#177776         ;PSW CORRECT?
4887 020306  001401                            BEQ     6$                      ;YES GO ON
4888 020310  104001                            ERROR   .1              ;CPU ERROR
4889                                                                   ;NO GO TO ERROR
4890 020312  000277                    6$:     SCC                             ;SETUP PSW
4891 020314  000253                            .WORD   253                     ; TEST CLN CLC CLV
4892 020316  022737  030004  177776            CMP     #30004,@#177776         ;PSW CORRECT?
4893 020324  001401                            BEQ     7$                      ;YES GO ON
4894 020326  104001                            ERROR   .1              ;CPU ERROR
4895                                                                   ;NO GO TO ERROR
4896 020330  000277                    7$:     SCC                             ;SETUP PSW
4897 020332  000254                            .WORD   254                     ; TEST CLN CLZ
4898 020334  022737  030003  177776            CMP     #30003,@#177776         ;PSW CORRECT?
4899 020342  001401                            BEQ     8$                      ;YES GO ON
4900 020344  104001                            ERROR   .1              ;CPU ERROR
4901                                                                   ;NO GO TO ERROR
4902 020346  000277                    8$:     SCC                             ;SETUP PSW
4903 020350  000255                            .WORD   255                     ; TEST CLN CLC CLZ
4904 020352  022737  030002  177776            CMP     #30002,@#177776         ;PSW CORRECT?
4905 020360  001401                            BEQ     9$                      ;YES GO ON
4906 020362  104001                            ERROR   .1              ;CPU ERROR
4907                                                                   ;NO GO TO ERROR
4908 020364  000277                    9$:     SCC                             ;SETUP PSW
4909 020366  000256                            .WORD   256                     ; TEST CLN CLV CLZ
4910 020370  022737  030001  177776            CMP     #30001,@#177776         ;SETUP PSW
4911 020376  001401                            BEQ     10$                     ;YES GO ON
4912 020400  104001                            ERROR   .1              ;CPU ERROR
4913                                                                   ;NO GO TO ERROR
4914 020402                            10$:
4915                                   ;
4918 020402                            TE113:
4919                                   ;       TEST MULTIPLE SETS OF CC BITS
4920 020402  012737  030000  177776            MOV     #30000,@#177776         ;INIT PSW
4921 020410  000263                            .WORD   263                     ; TEST SEC SEV
4922 020412  022737  030003  177776            CMP     #30003,@#177776         ;PSW CORRECT?
4923 020420  001401                            BEQ     1$                      ;YES GO ON
4924 020422  104001                            ERROR   .1              ;CPU ERROR
```

********** DOUBLE OPERAND TESTS **********

```
4925
4926 020424  000257                    1$:     CCC                             ;NO GO TO ERROR
4927 020426  000265                            .WORD   265                             ;SETUP PSW
4928 020430  022737  030005  177776            CMP     @#30005,@#177776                ; TEST SEC SEZ
4929 020436  001401                            BEQ     2$                              ;PSW CORRECT?
4930 020440  104001                            ERROR   .1                              ;YES GO ON
4931                                                                            ;CPU ERROR
4932 020442  000257                    2$:     CCC                             ;NO GO TO ERROR
4933 020444  000266                            .WORD   266                             ;SETUP PSW
4934 020446  022737  030006  177776            CMP     @#30006,@#177776                ; TEST SEV SEZ
4935 020454  001401                            BEQ     3$                              ;PSW CORRECT?
4936 020456  104001                            ERROR   .1                              ;YES GO ON
4937                                                                            ;CPU ERROR
4938 020460  000257                    3$:     CCC                             ;NO GO TO ERROR
4939 020462  000267                            .WORD   267                             ;SETUP PSW
4940 020464  022737  030007  177776            CMP     @#30007,@#177776                ; TEST SEC SEV SEZ
4941 020472  001401                            BEQ     4$                              ;PSW CORRECT?
4942 020474  104001                            ERROR   .1                              ;YES GO ON
4943                                                                            ;CPU ERROR
4944 020476  000257                    4$:     CCC                             ;NO GO TO ERROR
4945 020500  000271                            .WORD   271                             ;SETUP PSW
4946 020502  022737  030011  177776            CMP     @#30011,@#177776                ; TEST SEN SEC
4947 020510  001401                            BEQ     5$                              ;PSW CORRECT?
4948 020512  104001                            ERROR   .1                              ;YES GO ON
4949                                                                            ;CPU ERROR
4950 020514  000257                    5$:     CCC                             ;NO GO TO ERROR
4951 020516  000272                            .WORD   272                             ;SETUP PSW
4952 020520  022737  030012  177776            CMP     @#30012,@#177776                ; TEST SEN SEV
4953 020526  001401                            BEQ     6$                              ;PSW CORRECT?
4954 020530  104001                            ERROR   .1                              ;YES GO ON
4955                                                                            ;CPU ERROR
4956 020532  000257                    6$:     CCC                             ;NO GO TO ERROR
4957 020534  000273                            .WORD   273                             ;SETUP PSW
4958 020536  022737  030013  177776            CMP     @#30013,@#177776                ; TEST SEN SEC SEV
4959 020544  001401                            BEQ     7$                              ;PSW CORRECT?
4960 020546  104001                            ERROR   .1                              ;YES GO ON
4961                                                                            ;CPU ERROR
4962 020550  000257                    7$:     CCC                             ;NO GO TO ERROR
4963 020552  000274                            .WORD   274                             ;SETUP PSW
4964 020554  022737  030014  177776            CMP     @#30014,@#177776                ; TEST SEN SEZ
4965 020562  001401                            BEQ     8$                              ;PSW CORRECT?
4966 020564  104001                            ERROR   .1                              ;YES GO ON
4967                                                                            ;CPU ERROR
4968 020566  000257                    8$:     CCC                             ;NO GO TO ERROR
4969 020570  000275                            .WORD   275                             ;SETUP PSW
4970 020572  022737  030015  177776            CMP     @#30015,@#177776                ; TEST SEN SEC SEZ
4971 020600  001401                            BEQ     9$                              ;PSW CORRECT?
4972 020602  104001                            ERROR   .1                              ;YES GO ON
4973                                                                            ;CPU ERROR
4974 020604  000257                    9$:     CCC                             ;NO GO TO ERROR
4975 020606  000276                            .WORD   276                             ;SETUP PSW
4976 020610  022737  030016  177776            CMP     @#30016,@#177776                ; TEST SEN SEV SEZ
4977 020616  001401                            BEQ     10$                             ;PSW CORRECT?
4978 020620  104001                            ERROR   .1                              ;YES GO ON
4979                                                                            ;CPU ERROR
4980 020622                            10$:                                    ;NO GO TO ERROR
4981                                   ;
```

********** DOUBLE OPERAND TESTS **********

```
4984 020622                          TE113A:
4985                                 ;       TEST SIGNED AND CONDITIONAL BRANCHES
4986 020622  000257                          CCC                             ;CLEAR ALL CC BITS IN PSW
4987 020624  002001                          BGE     1$                      ;BGE SHOULD BRANCH
4988 020626  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; DIDN'T BRANCH
4989
4990 020630  003001                  1$:     BGT     2$                      ;BGT SHOULD BRANCH
4991 020632  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR;  DIDN'T BRANCH
4992
4993 020634  003401                  2$:     BLE     3$                      ;BLE SHOULDN'T BRANCH
4994 020636  000401                          BR      4$                      ;BRANCH TO NEXT TEST
4995 020640  104001                  3$:     ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLE SHOULD NOT HAVE BRANCHED
4996
4997 020642  002401                  4$:     BLT     5$                      ;BLT SHOULD NOT BRANCH
4998 020644  000401                          BR      6$                      ;BRANCH TO NEXT TEST
4999 020646  104001                  5$:     ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLT SHOULD NOT HAVE BRANCHED
5000
5001 020650  000264                  6$:     SEZ                             ;SET THE Z BIT IN PSW
5002 020652  003401                          BLE     7$                      ;BLE SHOULD BRANCH
5003 020654  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLE DIDN'T BRANCH
5004
5005 020656  003001                  7$:     BGT     8$                      ;BGT SHOULD NOT BRANCH
5006 020660  000401                          BR      9$                      ;BRANCH TO NEXT TEST
5007 020662  104001                  8$:     ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5008
5009 020664  000257                  9$:     CCC                             ;CLEAR ALL CC BITS IN PSW
5010 020666  000270                          SEN                             ;SET N BIT IN PSW
5011 020670  002401                          BLT     10$                     ;SHOULD BRANCH TO NEXT TEST
5012 020672  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLT SHOULD HAVE BRANCHED
5013
5014 020674  003401                  10$:    BLE     11$                     ;SHOULD BRANCH TO NEXT TEST
5015 020676  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLE SHOULD HAVE BRANCHED
5016
5017 020700  002001                  11$:    BGE     12$                     ;BGE SHOULD NOT BRANCH
5018 020702  000401                          BR      13$                     ;BRANCH TO NEXT TEST
5019 020704  104001                  12$:    ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5020
5021 020706  003001                  13$:    BGT     14$                     ;BGT SHOULD NOT BRANCH
5022 020710  000401                          BR      15$                     ;BRANCH TO NEXT TEST
5023 020712  104001                  14$:    ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5024
5025 020714  000257                  15$:    CCC                             ;CLEAR ALL CC BITS
5026 020716  000262                          SEV                             ;SET V BIT IN PSW
5027 020720  003401                          BLE     16$                     ;BLE SHOULD BRANCH
5028 020722  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLE DIDN'T BRANCH
5029
5030 020724  002401                  16$:    BLT     17$                     ;BLT SHOULD BRANCH
5031 020726  104001                          ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BLT DIDN'T BRANCH
5032
5033 020730  002001                  17$:    BGE     18$                     ;BGE SHOULDN'T BRANCH
5034 020732  000401                          BR      19$                     ;BRANCH TO NEXT TEST
5035 020734  104001                  18$:    ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5036
5037 020736  003001                  19$:    BGT     20$                     ;BGT SHOULD NOT BRANCH
5038 020740  000401                          BR      21$                     ;BRANCH TO NEXT TEST
5039 020742  104001                  20$:    ERROR   ;1              ;CPU ERROR
                                             ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5040
```

```
********** DOUBLE OPERAND TESTS ******************

 5041 020744  000257                  21$:    CCC                             ;CLEAR ALL CC BITS
 5042 020746  000272                          .WORD   272                     ;SET N AND V BITS IN PSW
 5043 020750  002001                          BGE     22$                     ;BGE SHOULD BRANCH
 5044 020752  104001                          ERROR   .1                  ;CPU ERROR
 5045                                                                     ;ERROR; BGE DIDN'T BRANCH
 5046 020754  003001                  22$:    BGT     23$                     ;BGT SHOULD BRANCH
 5047 020756  104001                          ERROR   .1                  ;CPU ERROR
 5048                                                                     ;ERROR; BGT DIDN'T BRANCH
 5049 020760  003401                  23$:    BLE     24$                     ;BLE SHOULDN'T BRANCH
 5050 020762  000401                          BR      25$                     ;BRANCH TO NEXT TEST
 5051 020764  104001                  24$:    ERROR   .1                  ;CPU ERROR
 5052                                                                     ;ERROR; BLE SHOULD NOT HAVE BRANCHED
 5053 020766  002401                  25$:    BLT     26$                     ;BLT SHOULD NOT BRANCH
 5054 020770  000401                          BR      27$                     ;BRANCH TO NEXT TEST
 5055 020772  104001                  26$:    ERROR   .1                  ;CPU ERROR
 5056                                                                     ;ERROR; BLT SHOULD NOT HAVE BRANCHED
 5057 020774                          27$:
 5060 020774                          TE114:
 5061                                 ;       TEST NOP INST
 5062 020774  012737  030000  177776          MOV     #30000,@#177776         ;INIT PSW
 5063 021002  012700  000001                  MOV     #1,R0                   ;INIT R0
 5064 021006  012701  000002                  MOV     #2,R1                   ;INIT R1
 5065 021012  012702  000003                  MOV     #3,R2                   ;INIT R2
 5066 021016  012703  000004                  MOV     #4,R3                   ;INIT R3
 5067 021022  012704  000005                  MOV     #5,R4                   ;INIT R4
 5068 021026  012705  000006                  MOV     #6,R5                   ;INIT R5
 5069 021032  010637  003012                  MOV     R6,@#SLOC00             ;SAVE SP
 5070 021036  012737  030017  111462          MOV     #30017,@#EXPDAT         ;SETUP PSW
 5071 021044  000277                          SCC                             ;SET ALL CONDITION CODE BITS
 5072 021046  000240                          NOP                             ; TEST INSTRUCTION
 5073 021050  000240                          NOP                             ; TEST INSTRUCTION
 5074 021052  000240                          NOP                             ; TEST INSTRUCTION
 5075 021054  004767  000046                  JSR     PC,T114                 ;CHECK PSW, AND GPR'S
 5076 021060  020637  003012                  CMP     R6,@#SLOC00             ;CHECK SP
 5077 021064  001401                          BEQ     1$                      ;OK GO ON
 5078 021066  104001                          ERROR   .1                  ;CPU ERROR
 5079                                                                     ;NO GO TO ERROR
 5080 021070  012737  030000  111462  1$:     MOV     #30000,@#EXPDAT         ;SETUP PSW
 5081 021076  000257                          CCC                             ;CLEAR ALL CONDITION CODE BITS
 5082 021100  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
 5083 021102  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
 5084 021104  000260                          .WORD   260                     ; TEST FOR NOP OPERATION
 5085 021106  004767  000014                  JSR     PC,T114                 ;CHECK PSW, AND GPR'S
 5086 021112  020637  003012                  CMP     R6,@#SLOC00             ;CHECK SP
 5087 021116  001401                          BEQ     2$                      ;OK GO ON
 5088 021120  104001                          ERROR   .1                  ;CPU ERROR
 5089                                                                     ;NO GO TO ERROR
 5090 021122                          2$:
 5091 021122  000167  000104                  JMP     FINNOP
 5092                                 ;
 5093 021126  023737  111462  177776  T114:   CMP     @#EXPDAT,@#177776       ;CHECK PSW
 5094 021134  001405                          BEQ     TA114                   ;OK GO ON
 5095 021136  010067  157236                  MOV     R0,400                  ;SAVE R0
 5096 021142  104001                          ERROR   .1                  ;CPU ERROR
 5097                                                                     ;NO GO TO ERROR
 5098 021144  016700  157230                  MOV     400,R0                  ;RESTORE R0
 5099 021150  022700  000001          TA114:  CMP     #1,R0                   ;CHECK R0
```

```
021154  001401                          BEQ     TB114                   ;OK GO ON
021156  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021160  022701  000002          TB114:  CMP     #2,R1                   ;CHECK R1
021164  001401                          BEQ     TC114                   ;OK GO ON
021166  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021170  022702  000003          TC114:  CMP     #3,R2                   ;CHECK R2
021174  001401                          BEQ     TD114                   ;OK GO ON
021176  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021200  022703  000004          TD114:  CMP     #4,R3                   ;CHECK R3
021204  001401                          BEQ     TF114                   ;OK GO ON
021206  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021210  022704  000005          TF114:  CMP     #5,R4                   ;CHECK R4
021214  001401                          BEQ     TG114                   ;OK GO ON
021216  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021220  022705  000006          TG114:  CMP     #6,R5                   ;CHECK R5
021224  001401                          BEQ     TH114                   ;OK GO ON
021226  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
021230  000207                  TH114:  RTS     PC                      ;RETURN
                                ;
021232  000240                  FINNOP: NOP
                                ;
                                ;
                                ;---------------------------------
                                ;TEST ATERNATE REGISTER SET
                                ;
021234  005000                          CLR     R0                      ;------CLEAR------
021236  005001                          CLR     R1                      ;-----PRIMARY----
021240  005002                          CLR     R2                      ;-----GENERAL-----
021242  005003                          CLR     R3                      ;-----PURPOSE-----
021244  005004                          CLR     R4                      ;-----REGISTER----
021246  005005                          CLR     R5                      ;-------SET-------
021250  012767  004000  156520          MOV     #4000,PS                ;SELECT ALTERNATE REGISTER SET
021256  032767  004000  156512          BIT     #BIT11,PS               ;TEST TO SEE THAT BIT IS SET IN PS
021264  001001                          BNE     1$                      ;IF IT'S SET GO TESTS REGISTERS
021266  104001                          ERROR   +1              ;CPU ERROR
                                                                ;ERROR! ALTERNATE REGISTER SELECT
                                                                        ;BIT IN PS IS NOT SET
021270  012700  177777          1$:     MOV     #177777,R0              ;WRITE ALL ONES TO ALTERNATE REG SET
021274  010001                          MOV     R0,R1                   ;
021276  010102                          MOV     R1,R2                   ;
021300  010203                          MOV     R2,R3                   ;
021302  010304                          MOV     R3,R4                   ;
021304  010405                          MOV     R4,R5                   ;
021306  042767  004000  156462          BIC     #BIT11,PS               ;CLEAR BIT 11 IN PS
021314  032767  004000  156454          BIT     #BIT11,PS               ;IS BIT 11 = 0?
021322  001401                          BEQ     2$                      ;IF IT'S NOT ZERO
021324  104001                          ERROR   +1              ;CPU ERROR
                                                                ;ERROR! BIT 11 DID NOT CLEAR
021326  005700                  2$:     TST     R0                      ;MAKE SURE THAT WE REALLY
021330  001401                          BEQ     3$                      ;WERE TALKING TO ALTERNATE REGISTERS.
021332  104001                          ERROR   +1              ;CPU ERROR
```

```
                                                                        ;ERROR!
021334  005701                          3$:     TST     R1                      ;
021336  001401                                  BEQ     4$
021340  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;ERROR!
021342  005702                          4$:     TST     R2                      ;
021344  001401                                  BEQ     5$                      ;
021346  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;ERROR!
021350  005703                          5$:     TST     R3                      ;
021352  001401                                  BEQ     6$                      ;
021354  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021356  005704                          6$:     TST     R4                      ;
021360  001401                                  BEQ     7$                      ;
021362  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021364  005705                          7$:     TST     R5                      ;
021366  001401                                  BEQ     8$                      ;
021370  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;ERROR!
021372  012767  004000  156376          8$:     MOV     #BIT11,PS               ;ENABLE ALTERNATE REGISTER SET
021400  005000                                  CLR     R0                      ;-----CLEAR-----
021402  005001                                  CLR     R1                      ;---ALTERNATE---
021404  005002                                  CLR     R2                      ;----REGISTER---
021406  005003                                  CLR     R3                      ;------SET-----
021410  005004                                  CLR     R4                      ;
021412  005005                                  CLR     R5                      ;
021414  042767  004000  156354                  BIC     #BIT11,PS               ;BACK TO PRIMARY GPRS
021422  012700  177777                          MOV     #177777,R0              ;ENSURE THAT WRITES TO PRIMARY GPRS
021426  010001                                  MOV     R0,R1                   ;DO NOT EFFECT ALTERNATE GPRS
021430  010102                                  MOV     R1,R2                   ;
021432  010203                                  MOV     R2,R3                   ;
021434  010304                                  MOV     R3,R4                   ;
021436  010405                                  MOV     R4,R5                   ;
021440  052767  004000  156330                  BIS     #BIT11,PS               ;RETURN TO ALTERNATE REGISTERS
021446  005700                                  TST     R0                      ;SHOULD BE ALL 0'S
021450  001401                                  BEQ     9$                      ;
021452  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;ERROR!
021454  005701                          9$:     TST     R1                      ;
021456  001401                                  BEQ     10$                     ;
021460  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021462  005702                          10$:    TST     R2                      ;
021464  001401                                  BEQ     11$                     ;
021466  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021470  005703                          11$:    TST     R3                      ;
021472  001401                                  BEQ     12$                     ;
021474  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021476  005704                          12$:    TST     R4                      ;
021500  001401                                  BEQ     13$                     ;
021502  104001                                  ERROR   +1              ;CPU ERROR
                                                                        ;
021504  005705                          13$:    TST     R5                      ;
```

```
********** DOUBLE OPERAND TESTS ******************

5216 021506  001401                        BEQ     14$                     ;
5217 021510  104001                        ERROR   +1              ;CPU ERROR
5218 021512                        14$:
5219
5220 021512                        ALROTS:
5221                               ;       ALTERNATE REGISTER SET R0 BIT TESTS
5222 021512  012700  177777                MOV     #177777,R0              ;R0=177777
5223 021516  020027  177777                CMP     R0,#177777              ;DOES R0=177777
5224 021522  001401                        BEQ     1$                      ;YES GO ON
5225 021524  104001                        ERROR   +1              ;CPU ERROR
5226                                                                       ;NO GO TO ERROR
5227 021526  005000                1$:     CLR     R0                      ;R0=0
5228 021530  020027  000000                CMP     R0,#0                   ;DOES R0=0
5229 021534  001401                        BEQ     2$                      ;YES GO ON
5230 021536  104001                        ERROR   +1              ;CPU ERROR
5231                                                                       ;NO GO TO ERROR
5232 021540  012700  125252        2$:     MOV     #125252,R0              ;R0=125252
5233 021544  020027  125252                CMP     R0,#125252              ;DOES R0=125252
5234 021550  001401                        BEQ     3$                      ;YES GO ON
5235 021552  104001                        ERROR   +1              ;CPU ERROR
5236                                                                       ;NO GO TO ERROR
5237 021554  012700  052525        3$:     MOV     #52525,R0               ;R0=52525
5238 021560  020027  052525                CMP     R0,#52525               ;DOES R0=52525
5239 021564  001401                        BEQ     4$                      ;YES GO ON
5240 021566  104001                        ERROR   +1              ;CPU ERROR
5241                                                                       ;NO GO TO ERROR
5242 021570                        4$:
5243                               ;
5246 021570                        ALR1TS:
5247                               ;       ALTERNATE REGISTER SET R1 BIT TESTS
5248 021570  012701  177777                MOV     #177777,R1              ;R1=177777
5249 021574  020127  177777                CMP     R1,#177777              ;DOES R1=177777
5250 021600  001401                        BEQ     1$                      ;YES GO ON
5251 021602  104001                        ERROR   +1              ;CPU ERROR
5252                                                                       ;NO GO TO ERROR
5253 021604  005001                1$:     CLR     R1                      ;R1=0
5254 021606  020127  000000                CMP     R1,#0                   ;DOES R1=0
5255 021612  001401                        BEQ     2$                      ;YES GO ON
5256 021614  104001                        ERROR   +1              ;CPU ERROR
5257                                                                       ;NO GO TO ERROR
5258 021616  012701  125252        2$:     MOV     #125252,R1              ;R1=125252
5259 021622  020127  125252                CMP     R1,#125252              ;DOES R1=125252
5260 021626  001401                        BEQ     3$                      ;YES GO ON
5261 021630  104001                        ERROR   +1              ;CPU ERROR
5262                                                                       ;NO GO TO ERROR
5263 021632  012701  052525        3$:     MOV     #52525,R1               ;R1=52525
5264 021636  020127  052525                CMP     R1,#52525               ;DOES R1=52525
5265 021642  001401                        BEQ     4$                      ;YES GO ON
5266 021644  104001                        ERROR   +1              ;CPU ERROR
5267                                                                       ;NO GO TO ERROR
5268 021646                        4$:
5269
5272 021646                        ALR2TS:
5273                               ;       ALTERNATE REGISTER SET R2 BIT TESTS
5274 021646  012702  177777                MOV     #177777,R2              ;R2=177777
5275 021652  020227  177777                CMP     R2,#177777              ;DOES R2=177777
5276 021656  001401                        BEQ     1$                      ;YES GO ON
```

```
********** DOUBLE OPERAND TESTS ******************

5277 021660  104001                        ERROR   +1              ;CPU ERROR
5278                                                                ;NO GO TO ERROR
5279 021662  005002                 1$:     CLR     R2              ;R2=0
5280 021664  020227  000000                CMP     R2,#0           ;DOES R2=0
5281 021670  001401                        BEQ     2$              ;YES GO ON
5282 021672  104001                        ERROR   +1              ;CPU ERROR
5283                                                                ;NO GO TO ERROR
5284 021674  012702  125252           2$:     MOV     #125252,R2      ;R2=125252
5285 021700  020227  125252                CMP     R2,#125252      ;DOES R2=125252
5286 021704  001401                        BEQ     3$              ;YES GO ON
5287 021706  104001                        ERROR   +1              ;CPU ERROR
5288                                                                ;NO GO TO ERROR
5289 021710  012702  052525           3$:     MOV     #52525,R2       ;R2=52525
5290 021714  020227  052525                CMP     R2,#52525       ;DOES R2=52525
5291 021720  001401                        BEQ     4$              ;YES GO ON
5292 021722  104001                        ERROR   +1              ;CPU ERROR
5293                                                                ;NO GO TO ERROR
5294 021724                         4$:
5295                                ;
5298 021724                         ALR3TS:
5299                                ;       ALTERNATE REGISTER SET R3 BIT TESTS
5300 021724  012703  177777                MOV     #177777,R3      ;R3=177777
5301 021730  020327  177777                CMP     R3,#177777      ;DOES R3=177777
5302 021734  001401                        BEQ     1$              ;YES GO ON
5303 021736  104001                        ERROR   +1              ;CPU ERROR
5304                                                                ;NO GO TO ERROR
5305 021740  005003                 1$:     CLR     R3              ;R3=0
5306 021742  020327  000000                CMP     R3,#0           ;DOES R3=0
5307 021746  001401                        BEQ     2$              ;YES GO ON
5308 021750  104001                        ERROR   +1              ;CPU ERROR
5309                                                                ;NO GO TO ERROR
5310 021752  012703  125252           2$:     MOV     #125252,R3      ;R3=125252
5311 021756  020327  125252                CMP     R3,#125252      ;DOES R3=125252
5312 021762  001401                        BEQ     3$              ;YES GO ON
5313 021764  104001                        ERROR   +1              ;CPU ERROR
5314                                                                ;NO GO TO ERROR
5315 021766  012703  052525           3$:     MOV     #52525,R3       ;R3=52525
5316 021772  020327  052525                CMP     R3,#52525       ;DOES R3=52525
5317 021776  001401                        BEQ     4$              ;YES GO ON
5318 022000  104001                        ERROR   +1              ;CPU ERROR
5319                                                                ;NO GO TO ERROR
5320 022002                         4$:
5321                                ;
5324 022002                         ALR4TS:
5325                                ;       ALTERNATE REGISTER SET R4 BIT TESTS
5326 022002  012704  177777                MOV     #177777,R4      ;R4=177777
5327 022006  020427  177777                CMP     R4,#177777      ;DOES R4=177777
5328 022012  001401                        BEQ     1$              ;YES GO ON
5329 022014  104001                        ERROR   +1              ;CPU ERROR
5330                                                                ;NO GO TO ERROR
5331 022016  005004                 1$:     CLR     R4              ;R4=0
5332 022020  020427  000000                CMP     R4,#0           ;DOES R4=0
5333 022024  001401                        BEQ     2$              ;YES GO ON
5334 022026  104001                        ERROR   +1              ;CPU ERROR
5335                                                                ;NO GO TO ERROR
5336 022030  012704  125252           2$:     MOV     #125252,R4      ;R4=125252
5337 022034  020427  125252                CMP     R4,#125252      ;DOES R4=125252
```

```
********** DOUBLE OPERAND TESTS ******************

 5338 022040  001401                          BEQ     3$                      ;YES GO ON
 5339 022042  104001                          ERROR   +1              ;CPU ERROR
 5340                                                                 ;NO GO TO ERROR
 5341 022044  012704  052525          3$:     MOV     #52525,R4               ;R4=52525
 5342 022050  020427  052525                  CMP     R4,#52525               ;DOES R4=52525
 5343 022054  001401                          BEQ     4$                      ;YES GO ON
 5344 022056  104001                          ERROR   +1              ;CPU ERROR
 5345                                                                 ;NO GO TO ERROR
 5346 022060                          4$:
 5347                                 ;
 5350 022060                          ALR5TS:
 5351                                 ;       ALTERNATE REGISTER SET R5 BIT TESTS
 5352 022060  012705  177777                  MOV     #177777,R5              ;R5=177777
 5353 022064  020527  177777                  CMP     R5,#177777              ;DOES R5=177777
 5354 022070  001401                          BEQ     1$                      ;YES GO ON
 5355 022072  104001                          ERROR   +1              ;CPU ERROR
 5356                                                                 ;NO GO TO ERROR
 5357 022074  005005          1$:             CLR     R5                      ;R5=0
 5358 022076  020527  000000                  CMP     R5,#0                   ;DOES R5=0
 5359 022102  001401                          BEQ     2$                      ;YES GO ON
 5360 022104  104001                          ERROR   +1              ;CPU ERROR
 5361                                                                 ;NO GO TO ERROR
 5362 022106  012705  125252          2$:     MOV     #125252,R5              ;R5=125252
 5363 022112  020527  125252                  CMP     R5,#125252              ;DOES R5=125252
 5364 022116  001401                          BEQ     3$                      ;YES GO ON
 5365 022120  104001                          ERROR   +1              ;CPU ERROR
 5366                                                                 ;NO GO TO ERROR
 5367 022122  012705  052525          3$:     MOV     #52525,R5               ;R5=52525
 5368 022126  020527  052525                  CMP     R5,#52525               ;DOES R5=52525
 5369 022132  001401                          BEQ     4$                      ;YES GO ON
 5370 022134  104001                          ERROR   +1              ;CPU ERROR
 5371                                                                 ;NO GO TO ERROR
 5372 022136  042767  004000  155632  4$:     BIC     #BIT11,PS               ;RETURN TO PRIMARY GEN PURPOSE REGS
 5373
 5374                                 ;
 5377 022144                          TE115:
 5378                                 ;       TEST MOVE FROM PROCCESSOR STATUS INST (MFPS)
 5379 022144  005004                          CLR     R4                      ;SETUP DESTINATION R4
 5380
 5381 022146  012701  022354                  MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
 5382 022152  012702  022364                  MOV     #TE115B,R2              ;
 5383 022156  012703  022372                  MOV     #TE115C,R3              ;
 5384 022162  012137  177776          1$:     MOV     (R1)+,@#177776          ;SETUP PSW
 5385 022166  106704                          MFPS    R4                      ; TEST INSTRUCTION
 5386 022170  023722  177776                  CMP     @#177776,(R2)+          ;CHECK PSW
 5387 022174  001401                          BEQ     2$                      ;OK GO ON
 5388 022176  104001                          ERROR   +1              ;CPU ERROR
 5389                                                                 ;NO GO TO ERROR
 5390 022200  020423          2$:             CMP     R4,(R3)+                ;CHECK R4
 5391 022202  001401                          BEQ     3$                      ;OK GO ON
 5392 022204  104001                          ERROR   +1              ;CPU ERROR
 5393                                                                 ;NO GO TO ERROR
 5394 022206  021127  177777          3$:     CMP     (R1),#177777            ;ARE WE DONE
 5395 022212  001363                          BNE     1$                      ;NO GO TO 1$
 5396
 5397
 5398 022214  012701  022354                  MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
```

```
********** DOUBLE OPERAND TESTS ******************

5399 022220  012702  022364                MOV     #TE115B,R2              ;
5400 022224  012703  022372                MOV     #TE115C,R3              ;
5401 022230  010605                        MOV     R6,R5                   ;SAVE STACK IN R5
5402 022232  011137  177776          101$: MOV     (R1),@#177776           ;SETUP PSW
5403 022236  106706                        MFPS    R6                      ; TEST INSTRUCTION
5404 022240  023712  177776                CMP     @#177776,(R2)           ;CHECK PSW
5405 022244  001401                        BEQ     102$                    ;OK GO ON
5406 022246  104001                        ERROR   +1              ;CPU ERROR
5407                                                               ;NO GO TO ERROR
5408 022250  020613                  102$: CMP     R6,(R3)                 ;CHECK R6
5409 022252  001401                        BEQ     103$                    ;OK GO ON
5410 022254  104001                        ERROR   +1              ;CPU ERROR
5411                                                               ;NO GO TO ERROR
5412 022256  010506                  103$: MOV     R5,R6                   ;RESTORE STACK
5413
5414
5415 022260  012701  022354                MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
5416 022264  012702  022364                MOV     #TE115B,R2              ;
5417 022270  012703  022400                MOV     #TE115D,R3              ;
5418 022274  005037  111462                CLR     @#EXPDAT                ;INIT EXPECTED DATA HOLDER.
5419 022300  012704  111462          4$:   MOV     #EXPDAT,R4              ;SETUP POINTER TO TEST LOCATION
5420 022304  012137  177776                MOV     (R1)+,@#177776          ;SETUP PSW
5421 022310  106724                        MFPS    (R4)+                   ; TEST INSTRUCTION
5422 022312  023722  177776                CMP     @#177776,(R2)+          ;CHECK PSW
5423 022316  001401                        BEQ     5$                      ;OK GO ON
5424 022320  104001                        ERROR   +1              ;CPU ERROR
5425                                                               ;NO GO TO ERROR
5426 022322  020427  111463          5$:   CMP     R4,#EXPDAT+1            ;CHECK R4
5427 022326  001401                        BEQ     6$                      ;OK GO ON
5428 022330  104001                        ERROR   +1              ;CPU ERROR
5429                                                               ;NO GO TO ERROR
5430 022332  023723  111462          6$:   CMP     @#EXPDAT,(R3)+          ;CHECK TEST LOCATION
5431 022336  001401                        BEQ     7$                      ;OK GO ON
5432 022340  104001                        ERROR   +1              ;CPU ERROR
5433                                                               ;NO GO TO ERROR
5434 022342  021127  177777          7$:   CMP     (R1),#177777            ;ARE WE DONE
5435 022346  001354                        BNE     4$                      ;NO GO TO 4$
5436
5437 022350  000167  000032                JMP     TE115F
5438
5439                                 ;
5440                                 ;
5441 022354  030207                TE115A: .WORD   30207
5442 022356  030000                        .WORD   30000
5443 022360  030057                        .WORD   30057
5444 022362  177777                        .WORD   177777
5445 022364  030211                TE115B: .WORD   30211
5446 022366  030004                        .WORD   30004
5447 022370  030041                        .WORD   30041
5448 022372  177607                TE115C: .WORD   177607
5449 022374  000000                        .WORD   0
5450 022376  000057                        .WORD   57
5451 022400  000207                TE115D: .WORD   207
5452 022402  000000                        .WORD   0
5453 022404  000057                        .WORD   57
5454 022406  000240                TE115F: NOP
5455                                 ;
```

```
********** DOUBLE OPERAND TESTS ******************

5458 022410                            TE116:
5459                                   ;       TEST MOVE TO PROCESSOR STATUS INST (MTPS)
5460
5461 022410  012737  030000  177776            MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5462 022416  012701  022712                      MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5463 022422  012702  022670                      MOV     #TE116B,R2              ;
5464 022426  010103                              MOV     R1,R3                   ;
5465 022430  004767  000136                      JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5466                                                                             ;AND SOURCE OPERAND
5467
5468
5469 022434  012737  140000  177776            MOV     #140000,@#177776        ;SET PSW TO USER MODE
5470 022442  012706  000600                      MOV     #600,R6                 ;SETUP USER STACK
5471 022446  012701  022712                      MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5472 022452  012702  022702                      MOV     #TE116C,R2              ;
5473 022456  010103                              MOV     R1,R3                   ;
5474 022460  004767  000106                      JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5475                                                                             ;AND SOURCE OPERAND
5476
5477
5478 022464  012737  140000  177776            MOV     #140000,@#177776        ;SET PSW TO USER MODE AND CLEAR CC BITS
5479 022472  012701  022664                      MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5480 022476  012702  022702                      MOV     #TE116C,R2              ;
5481 022502  012703  022712                      MOV     #TE116D,R3              ;
5482 022506  010104                              MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5483 022510  004767  000110                      JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5484                                                                             ;AND SOURCE OPERAND
5485
5486
5487 022514  012737  030000  177776            MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5488 022522  012701  022664                      MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5489 022526  012702  022670                      MOV     #TE116B,R2              ;
5490 022532  012703  022712                      MOV     #TE116D,R3              ;
5491 022536  010104                              MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5492 022540  004767  000060                      JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5493                                                                             ;AND SOURCE OPERAND
5494
5495 022544  005037  177776                      CLR     @#177776                ;SET PSW TO KERNEL MODE
5496 022550  106427  177412                      MTPS    #177412                 ;TEST INSTRUCTION
5497 022554  022737  000012  177776              CMP     #12,@#177776            ;IS PSW CORRECT
5498 022562  001401                              BEQ     100$                    ;YES GO ON
5499 022564  104001                              ERROR   .1              ;CPU ERROR
5500                                                                     ;NO GO TO ERROR
5501 022566                            100$:
5502 022566  000167  000130                      JMP     FIN116
5503                                   ;
5504 022572  012105                    T116:     MOV     (R1)+,R5                ;MOVE TEST DATA TO R5
5505 022574  106405                              MTPS    R5                      ; TEST INSTRUCTION
5506 022576  023722  177776                      CMP     @#177776,(R2)+          ;IS PSW CORRECT
5507 022602  001401                              BEQ     1$                      ;YES GO ON
5508 022604  104001                              ERROR   .1              ;CPU ERROR
5509                                                                     ;NO GO TO ERROR
5510 022606  022305                    1$:       CMP     (R3)+,R5                ;IS R5 CORRECT
5511 022610  001401                              BEQ     2$                      ;YES GO ON
5512 022612  104001                              ERROR   .1              ;CPU ERROR
5513                                                                     ;NO GO TO ERROR
5514 022614  021227  177777            2$:       CMP     (R2),#177777            ;ARE WE DONE
```

```
********** DOUBLE OPERAND TESTS ******************

5515 022620  001364                        BNE     T116                    ;NO GO TO T116
5516 022622  000207                        RTS     PC                      ;RETURN
5517                                ;
5518 022624  106421                TA116:  MTPS    (R1)+                   ; TEST INSTRUCTION
5519 022626  023722  177776                CMP     @#177776,(R2)+          ;IS PSW CORRECT
5520 022632  001401                        BEQ     1$                      ;YES GO ON
5521 022634  104001                        ERROR   +1              ;CPU ERROR
5522                                                                ;NO GO TO ERROR
5523 022636  122423                1$:     CMPB    (R4)+,(R3)+             ;CHECK TEST LOCATION
5524 022640  001401                        BEQ     2$                      ;OK GO ON
5525 022642  104001                        ERROR   +1              ;CPU ERROR
5526                                                                ;NO GO TO ERROR
5527 022644  020104                2$:     CMP     R1,R4                   ;IS SOURCE OPERAND CORRECT
5528 022646  001401                        BEQ     3$                      ;YES GO ON
5529 022650  104001                        ERROR   +1              ;CPU ERROR
5530                                                                ;NO GO TO ERROR
5531 022652  005203                3$:     INC     R3                      ;POINT TO NEXT WORD
5532 022654  021227  177777                CMP     (R2),#177777            ;ARE WE DONE
5533 022660  001361                        BNE     TA116                   ;NO GO TO TA116
5534 022662  000207                        RTS     PC                      ;RETURN
5535                                ;
5536 022664     377                TE116A: .BYTE   377
5537 022665     000                        .BYTE   0
5538 022666     252                        .BYTE   252
5539 022667     125                        .BYTE   125
5540 022670  030357                TE116B: .WORD   30357
5541 022672  030000                        .WORD   30000
5542 022674  030252                        .WORD   30252
5543 022676  030105                        .WORD   30105
5544 022700  177777                        .WORD   177777
5545 022702  140017                TE116C: .WORD   140017
5546 022704  140000                        .WORD   140000
5547 022706  140012                        .WORD   140012
5548 022710  140005                        .WORD   140005
5549 022712  177777                TE116D: .WORD   177777
5550 022714  177400                        .WORD   177400
5551 022716  177652                        .WORD   177652
5552 022720  177525                        .WORD   177525
5553                                ;
5554 022722                        FIN116:
5555                                ;
5565 022722                        TE117:
5566                                ;       TEST MFPT (MOVE FROM PROCESSOR TYPE)
5567 022722  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTOR
5568 022726  012737  023034  000010        MOV     #TE117A,@#10            ;SETUP VECTOR TO HANDLE POSSIBLE ILLEGAL INST TRAP
5569 022734  012700  177777                MOV     #177777,R0              ;INIT R0
5570 022740  012737  030000  177776        MOV     #30000,@#177776         ;SETUP PSW
5571 022746  000007                        .WORD   7                       ; TEST INSTRUCTION
5572 022750  022737  030000  177776        CMP     #30000,@#177776         ;IS PSW CORRECT
5573 022756  001401                        BEQ     1$                      ;YES GO ON
5574 022760  104001                        ERROR   +1              ;CPU ERROR
5575                                                                ;NO GO TO ERROR
5576 022762  020027  000005        1$:     CMP     R0,#5                   ;IS R0 CORRECT
5577 022766  001401                        BEQ     2$                      ;YES GO ON
5578 022770  104001                        ERROR   +1              ;CPU ERROR
5579                                                                ;NO GO TO ERROR
5580 022772  012700  177777        2$:     MOV     #177777,R0              ;INIT R0
```

```
********** DOUBLE OPERAND TESTS **********

5581 022776  000277                        SCC                             ;SET ALL CC BITS
5582 023000  000007                        .WORD   7                       ; TEST INSTRUCTION
5583 023002  022737  030017  177776        CMP     #30017,@#177776         ;IS PSW CORRECT
5584 023010  001401                        BEQ     3$                      ;YES GO ON
5585 023012  104001                        ERROR   ^1              ;CPU ERROR
5586                                                               ;NO GO TO ERROR
5587 023014  020027  000005          3$:   CMP     R0,#5                   ;IS R0 CORRECT
5588 023020  001401                        BEQ     4$                      ;YES GO ON
5589 023022  104001                        ERROR   ^1              ;CPU ERROR
5590                                                               ;NO GO TO ERROR
5591 023024  012637  000010          4$:   MOV     (SP)+,@#10              ;RESTORE VECTOR
5592
5593 023030  000167  000002                JMP     FIN117
5594                                ;
5595 023034  104001                TE117A: ERROR   ^1              ;CPU ERROR
5596                                                               ;GO TO ERROR IF TRAP TAKES PLACE
5597                                ;
5598 023036  000240                FIN117: NOP
5599                                ;
5611 023040                        TE120:
5612                                ;       TEST HALT (NOT KERNEL MODE)
5613 023040  005037  177766                CLR     @#177766                ;INIT CPU ERROR REG
5614 023044  005037  177776                CLR     @#177776                ;INIT PSW-SET KERNEL MODE
5615 023050  013746  000004                MOV     @#4,-(SP)               ;SAVE VECTOR
5616 023054  013746  000006                MOV     @#6,-(SP)               ;SAVE VECTOR
5617 023060  012737  023114  000004        MOV     #TE120A,@#4             ;SET UP VECTOR TO HANDLE ILLEGAL HALT
5618 023066  005037  000006                CLR     @#6                     ;SET UP VECTOR TO COME BACK IN KERNEL MODE
5619 023072  012767  140000  154676        MOV     #140000,PS              ;SET IN USER MODE
5620 023100  012706  000600                MOV     #600,R6                 ;INITIALIZE THE USER STACK POINTER
5621 023104  000000                        HALT                            ; TEST INSTRUCTION
5622 023106  104001                PROCNT: ERROR   ^1              ;CPU ERROR
5623                                                               ;IF NOTHING HAPPENED GO TO ERROR
5624 023110  000167  000044                JMP     FIN120                  ;
5625                                ;
5626                                ;
5627 023114  022737  030000  177776  TE120A: CMP     #30000,@#177776         ;IS PSW CORRECT/PREVIOUS MODE = USER?
5628 023122  001401                        BEQ     1$                      ;YES GO ON
5629 023124  104001                        ERROR   ^1              ;CPU ERROR
5630                                                               ;NO GO TO ERROR
5631 023126  022737  000200  177766  1$:   CMP     #200,@#177766           ; TEST CPU ERROR REGISTER
5632 023134  001401                        BEQ     2$                      ;YES GO ON
5633 023136  104001                        ERROR   ^1              ;CPU ERROR
5634                                                               ;NO GO TO ERROR
5635 023140  022627  023106          2$:   CMP     (SP)+,#PROCNT           ;DOES STACK CONTAIN CORRECT PC
5636 023144  001401                        BEQ     3$                      ;YES GO ON
5637 023146  104001                        ERROR   ^1              ;CPU ERROR
5638                                                               ;NO GO TO ERROR
5639 023150  022627  140000          3$:   CMP     (SP)+,#140000           ;DOES STACK CONTAIN CORRECT PSW
5640 023154  001401                        BEQ     FIN120                  ;YES GO ON
5641 023156  104001                        ERROR   ^1              ;CPU ERROR
5642                                                               ;NO GO TO ERROR
5643 023160  012637  000006        FIN120: MOV     (SP)+,@#6               ;RESTORE VECTOR
5644 023164  012637  000004                MOV     (SP)+,@#4               ;RESTORE VECTOR
5645
5646                                ;
5649 023170                        TE121:
5650                                ;       TEST RESET
```

```
********** DOUBLE OPERAND TESTS **********

5651                                    ;       CMPB    @APTENV,$ENV            ;ARE WE IN APT MODE?
5652                                    ;       BNE     1$                      ;IF NOT: DO THIS TEST
5653 023170  005767  156012                     TST     $PASS                   ;FIRST PASS??
5654 023174  001402                             BEQ     1$                      ;IF YES, DO IT
5655 023176  000167  000622                     JMP     FIN122                  ;ELSE SKIP THIS TEST BECAUSE RESETS
5656                                                                            ;SCREW UP THE APT MONITOR.
5657 023202  012737  030340  177776  1$:       MOV     #30340,@#177776         ;SETUP PSW TO KERNEL MODE
5658 023210  012737  160000  177572            MOV     #160000,@#177572        ;SETUP MMR0
5659 023216  012737  000077  172516            MOV     #77,@#172516            ;SETUP MMR3
5660 023224  005037  177772                     CLR     @#177772                ;CLEAR PIRQ
5661 023230  023727  177772  000000            CMP     @#177772,#0             ;IS PIRQ CORRECT
5662 023236  001401                             BEQ     C121A                   ;YES GO ON
5663 023240  104001                             ERROR   ^1              ;CPU ERROR
5664                                                                    ;NO GO TO ERROR
5665 023242  012737  025000  177772  C121A:    MOV     #25000,@#177772         ;MOVE AN ALTERNATING PATTERN TO PIRQ
5666 023250  022737  025252  177772            CMP     #25252,@#177772         ;IS PIRQ CORRECT
5667 023256  001401                             BEQ     C121B                   ;YES GO ON
5668 023260  104001                             ERROR   ^1              ;CPU ERROR
5669                                                                    ;NO GO TO ERROR
5670 023262  012737  077000  177772  C121B:    MOV     #77000,@#177772         ;SETUP PIRQ
5671 023270  022737  077314  177772            CMP     #77314,@#177772         ;IS PIRQ CORRECT
5672 023276  001401                             BEQ     C121C                   ;YES GO ON
5673 023300  104001                             ERROR   ^1              ;CPU ERROR
5674                                                                    ;NO GO TO ERROR
5675 023302  000277                     C121C:  SCC                             ;SET ALL CC BITS
5676 023304  000005                             RESET                           ; TEST INSTRUCTION
5677 023306  022737  030357  177776            CMP     #30357,@#177776         ;IS PSW CORRECT
5678 023314  001401                             BEQ     1$                      ;YES GO ON
5679 023316  104001                             ERROR   ^1              ;CPU ERROR
5680                                                                    ;NO GO TO ERROR
5681 023320  013701  177572            1$:     MOV     @#SR0,R1                ;SAVE SR0 IN R1.
5682 023324  042701  000176                     BIC     #176,R1                 ;STRIP OFF UNDEFINED BITS 1-6 FROM MMR0
5683 023330  022701  000000                     CMP     #0,R1                   ;IS MMR0 CORRECT
5684 023334  001401                             BEQ     2$                      ;YES GO ON
5685 023336  104001                             ERROR   ^1              ;CPU ERROR
5686                                                                    ;NO GO TO ERROR
5687 023340  022737  000000  172516  2$:       CMP     #0,@#172516             ;IS MMR3 CORRECT
5688 023346  001401                             BEQ     3$                      ;YES GO ON
5689 023350  104001                             ERROR   ^1              ;CPU ERROR
5690                                                                    ;NO GO TO ERROR
5691 023352  022737  000000  177772  3$:       CMP     #0,@#177772             ;IS PIRQ CORRECT
5692 023360  001401                             BEQ     4$                      ;YES GO ON
5693 023362  104001                             ERROR   ^1              ;CPU ERROR
5694                                                                    ;NO GO TO ERROR
5695
5696 023364  013702  000004            4$:     MOV     @#4,R2                  ;SAVE LOC 4 IN R2
5697 023370  013703  000006                     MOV     @#6,R3                  ;SAVE LOC 6 IN R3
5698 023374  012737  023532  000004             MOV     #8$,@#4                 ;SETUP VECTORS IN CASE AN ERROR CAUSES
5699                                                                            ;A HALT TO OCCUR IN USER MODE.
5700 023402  012737  000340  000006             MOV     #340,@#6                ;THIS SETS KERNEL MODE, AND PREVENTS PRIQ
5701                                                                            ;INTERRUPTS FROM TRASHING THE STACK IF A
5702                                                                            ;USER MODE HALT OCCURS.
5703 023410  012737  140340  177776             MOV     #140340,@#177776        ;SETUP PSW TO USER MODE
5704 023416  012737  160000  177572             MOV     #160000,@#177572        ;SETUP MMR0
5705 023424  012737  000077  172516             MOV     #77,@#172516            ;SETUP MMR3
5706 023432  012737  077000  177772             MOV     #77000,@#177772         ;SETUP PIRQ
5707 023440  000277                             SCC                             ;SET ALL CC BITS
```

```
023442  000005                          RESET                           ; TEST INSTRUCTION
023444  022737  140357  177776          CMP     #140357,@#177776        ;IS PSW CORRECT
023452  001402                          BEQ     5$                      ;YES GO ON
023454  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
023456  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023460  013701  177572          5$:     MOV     @#177572,R1             ;SAVE MMR0 IN R1
023464  042701  000176                  BIC     #176,R1                 ;CLEAR BITS 1-6 FROM MMR0
023470  022701  160000                  CMP     #160000,R1              ;IS MMR0 CORRECT
023474  001402                          BEQ     6$                      ;YES GO ON
023476  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
023500  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023502  022737  000077  172516  6$:     CMP     #77,@#172516            ;IS MMR3 CORRECT
023510  001402                          BEQ     7$                      ;YES GO ON
023512  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
023514  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023516  022737  077314  177772  7$:     CMP     #77314,@#177772         ;IS PIRQ CORRECT
023524  001403                          BEQ     9$                      ;YES GO ON
023526  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
023530  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023532  000002                  8$:     RTI                             ;USER MODE HALT OCCURRED; GO TO ERROR.

023534  005037  177772          9$:     CLR     @#177772                ;CLEAR PIRQ
023540  005037  177776                  CLR     @#177776                ;CLEAR PSW
023544  010237  000004                  MOV     R2,@#4                  ;RESTORE VECTORS TO PREVIOUS STATE
023550  010337  000006                  MOV     R3,@#6                  ;
023554                          FIN121:
                                ;
023554                          TE122:
                                ;       TEST SPL (SET PRIORITY LEVEL)

023554  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
023560  012701  024004                  MOV     #T122B,R1               ;SETUP POINTER TO DATA
023564  012737  030000  177776  1$:     MOV     #30000,@#177776         ;INIT PSW
023572  004767  000054                  JSR     PC,T122A                ; TEST INSTRUCTION
023576  022137  177776                  CMP     (R1)+,@#177776          ;IS PSW CORRECT
023602  001401                          BEQ     2$                      ;YES GO ON
023604  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023606  077512                  2$:     SOB     R5,1$                   ;REPEAT UNTIL ALL CASES ARE TESTED


023610  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
023614  012737  140000  177776  3$:     MOV     #140000,@#177776        ;SETUP PSW TO USER MODE
023622  012706  000600                  MOV     #600,R6                 ;SETUP USER STACK
023626  004767  000020                  JSR     PC,T122A                ; TEST INSTRUCTION
023632  022737  140017  177776          CMP     #140017,@#177776        ;IS PSW CORRECT
023640  001401                          BEQ     4$                      ;YES GO ON
023642  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
023644  077515                  4$:     SOB     R5,3$                   ;REPEAT UNTIL ALL CASES ARE TESTED


023646  000167  000152                  JMP     FIN122
```

********** DOUBLE OPERAND TESTS **********

```
5767                                    ;
5768 023652  020527  000010            T122A:  CMP     R5,#8.                  ;FIND OUT WHAT COUNTER IS
5769 023656  001003                            BNE     1$                      ;IF NOT PRIORITY 0 GO TO 1$
5770 023660  000277                            SCC                             ;SET ALL CC BITS
5771 023662  000230                            SPL     0                       ;SET PRIORITY TO 0
5772 023664  000446                            BR      8$                      ;RETURN
5773 023666  020527  000007            1$:     CMP     R5,#7                   ;FIND OUT WHAT COUNTER IS
5774 023672  001003                            BNE     2$                      ;IF NOT PRIORITY 1 GO TO 2$
5775 023674  000277                            SCC                             ;SET ALL CC BITS
5776 023676  000231                            SPL     1                       ;SET PRIORITY TO 1
5777 023700  000440                            BR      8$                      ;RETURN
5778 023702  020527  000006            2$:     CMP     R5,#6                   ;FIND OUT WHAT COUNTER IS
5779 023706  001003                            BNE     3$                      ;IF NOT PRIORITY 2 GO TO 3$
5780 023710  000277                            SCC                             ;SET ALL CC BITS
5781 023712  000232                            SPL     2                       ;SET PRIORITY TO 2
5782 023714  000432                            BR      8$                      ;RETURN
5783 023716  020527  000005            3$:     CMP     R5,#5                   ;FIND OUT WHAT COUNTER IS
5784 023722  001003                            BNE     4$                      ;IF NOT PRIORITY 3 GO TO 4$
5785 023724  000277                            SCC                             ;SET ALL CC BITS
5786 023726  000233                            SPL     3                       ;SET PRIORITY TO 3
5787 023730  000424                            BR      8$                      ;RETURN
5788 023732  020527  000004            4$:     CMP     R5,#4                   ;FIND OUT WHAT COUNTER IS
5789 023736  001003                            BNE     5$                      ;IF NOT PRIORITY 4 GO TO 5$
5790 023740  000277                            SCC                             ;SET ALL CC BITS
5791 023742  000234                            SPL     4                       ;SET PRIORITY TO 4
5792 023744  000416                            BR      8$                      ;RETURN
5793 023746  020527  000003            5$:     CMP     R5,#3                   ;FIND OUT WHAT COUNTER IS
5794 023752  001003                            BNE     6$                      ;IF NOT PRIORITY 5 GO TO 6$
5795 023754  000277                            SCC                             ;SET ALL CC BITS
5796 023756  000235                            SPL     5                       ;SET PRIORITY TO 5
5797 023760  000410                            BR      8$                      ;RETURN
5798 023762  020527  000002            6$:     CMP     R5,#2                   ;FIND OUT WHAT COUNTER IS
5799 023766  001003                            BNE     7$                      ;IF NOT PRIORITY 6 GO TO 7$
5800 023770  000277                            SCC                             ;SET ALL CC BITS
5801 023772  000236                            SPL     6                       ;SET PRIORITY TO 6
5802 023774  000402                            BR      8$                      ;RETURN
5803 023776  000277                    7$:     SCC                             ;SET ALL CC BITS
5804 024000  000237                            SPL     7                       ;SET PRIORITY TO 7
5805 024002  000207                    8$:     RTS     PC                      ;RETURN
5806                                    ;
5807 024004  030017                    T122B:  .WORD   30017
5808 024006  030057                            .WORD   30057
5809 024010  030117                            .WORD   30117
5810 024012  030157                            .WORD   30157
5811 024014  030217                            .WORD   30217
5812 024016  030257                            .WORD   30257
5813 024020  030317                            .WORD   30317
5814 024022  030357                            .WORD   30357
5815 024024  000240                    FIN122: NOP
5816                                    ;
5819 024026                            TE123:
5820                                    ;       TEST TSTSET INSTRUCTION (MULTI PROCESSING INST)
5821 024026  005037  177776                    CLR     @#177776                ;INIT PSW
5822 024032  012703  000012                    MOV     #10.,R3                 ;INIT COUNTER
5823 024036  012701  000400                    MOV     #400,R1                 ;SETUP DESTINATION
5824 024042  012700  024210                    MOV     #T123A,R0               ;SETUP SOURCE
5825 024046  012021                    100$:   MOV     (R0)+,(R1)+             ;RELOCATE TABLES
```

```
024050  077302                          SOB     R3,100$                 ;ARE WE DONE
024052  013746  000010                  MOV     @#10,-(SP)              ;SAVE VECTOR
024056  012737  024234  000010          MOV     #T123D,@#10             ;SETUP NEW VECTOR
024064  005000                          CLR     R0                      ;INIT R0
024066  012701  000400                  MOV     #400,R1                 ;SETUP POINTERS TO TABLES
024072  012702  000410                  MOV     #410,R2                 ;
024076  012703  000416                  MOV     #416,R3                 ;
024102  010104                          MOV     R1,R4                   ;
024104  012737  030000  177776  1$:     MOV     #30000,@#177776         ;SETUP PSW
024112  000262                          SEV                             ;SET V BIT
024114  007221                          .WORD   7221                    ; TEST INSTRUCTION
024116  022237  177776                  CMP     (R2)+,@#177776          ;IS PSW CORRECT
024122  001401                          BEQ     2$                      ;YES GO ON
024124  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
024126  020013                  2$:     CMP     R0,(R3)                 ;IS R0 CORRECT
024130  001401                          BEQ     3$                      ;YES GO ON
024132  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
024134  005204                  3$:     INC     R4                      ;SETUP EXPECTED DATA
024136  005204                          INC     R4                      ;
024140  020401                          CMP     R4,R1                   ;IS R1 CORRECT
024142  001401                          BEQ     4$                      ;YES GO ON
024144  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
024146  052713  000001          4$:     BIS     #1,(R3)                 ;SETUP EXPECTED DATA
024152  022341                          CMP     (R3)+,-(R1)             ;IS TEST LOCATION CORRECT
024154  001401                          BEQ     5$                      ;YES GO ON
024156  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
024160  005201                  5$:     INC     R1                      ;POINT TO NEXT TEST LOCATION
024162  005201                          INC     R1                      ;
024164  021127  177777                  CMP     (R1),#177777            ;ARE WE DONE
024170  001345                          BNE     1$                      ;NO GO TO 1$
024172  012737  024236  000010          MOV     #T123E,@#10             ;SETUP NEW VECTOR
024200  007201                          .WORD   7201                    ; TEST INSTRUCTION ILLEGAL MODE
024202  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;GO TO ERROR IF DIDN'T TRAP
024204  000167  000032                  JMP     T123F
                                ;
                                ;
024210  167604                  T123A:  .WORD   167604
024212  000000                          .WORD   0
024214  000001                          .WORD   1
024216  177777                          .WORD   177777
024220  030010                  T123B:  .WORD   30010
024222  030004                          .WORD   30004
024224  030001                          .WORD   30001
024226  167604                  T123C:  .WORD   167604
024230  000000                          .WORD   0
024232  000001                          .WORD   1
024234  104001                  T123D:  ERROR   +1                ;CPU ERROR
                                                                  ;GO TO ERROR IF TRAPPED
024236  005726                  T123E:  TST     (SP)+                   ;CLEAN UP STACK
024240  005726                          TST     (SP)+                   ;
024242  012637  000010          T123F:  MOV     (SP)+,@#10              ;RESTORE VECTOR
```

```
********** DOUBLE OPERAND TESTS ******************

5883                                    ;
5886 024246                             TE124:
5887                                    ;       TEST WRTLCK (WRITE LOCK MULTI PROCESSING INST)
5888 024246  005037  177776                        CLR     @#177776                ;INIT PSW
5889 024252  012703  000012                        MOV     #10.,R3                 ;INIT COUNTER
5890 024256  012701  000400                        MOV     #400,R1                 ;SETUP DESTINATION
5891 024262  012700  024442                        MOV     #T124A,R0               ;SETUP SOURCE
5892 024266  012021                     100$:      MOV     (R0)+,(R1)+             ;RELOCATE TABLES
5893 024270  077302                                SOB     R3,100$                 ;ARE WE DONE
5894 024272  013746  000010                        MOV     @#10,-(SP)              ;SAVE VECTOR
5895 024276  012737  024466  000010                MOV     #T124D,@#10             ;SETUP NEW VECTOR
5896 024304  012701  000400                        MOV     #400,R1                 ;SETUP POINTERS TO TABLES
5897 024310  012702  000410                        MOV     #410,R2                 ;
5898 024314  012703  000416                        MOV     #416,R3                 ;
5899 024320  010204                                MOV     R2,R4                   ;
5900 024322  012737  030000  177776  1$:           MOV     #30000,@#177776         ;SETUP PSW
5901 024330  011100                                MOV     (R1),R0                 ;SETUP R0
5902 024332  020327  000416                        CMP     R3,#416                 ;IS THIS THE FIRST TEST CASE
5903 024336  001401                                BEQ     2$                      ;YES GO TO 2$
5904 024340  000402                                BR      3$                      ;NO GO TO 3$
5905 024342  000261                     2$:        SEC                             ;SET C BIT
5906 024344  000401                                BR      4$                      ;
5907 024346  000241                     3$:        CLC                             ;CLEAR C BIT
5908 024350  000262                     4$:        SEV                             ;SET V BIT
5909 024352  007322                                .WORD   7322                    ; TEST INSTRUCTION
5910 024354  022337  177776                        CMP     (R3)+,@#177776          ;IS PSW CORRECT
5911 024360  001401                                BEQ     5$                      ;YES GO ON
5912 024362  104001                                ERROR   +1              ;CPU ERROR
5913                                                                       ;NO GO TO ERROR
5914 024364  021100                     5$:        CMP     (R1),R0                 ;IS R0 CORRECT
5915 024366  001401                                BEQ     6$                      ;YES GO ON
5916 024370  104001                                ERROR   +1              ;CPU ERROR
5917                                                                       ;NO GO TO ERROR
5918 024372  005204                     6$:        INC     R4                      ;SETUP EXPECTED DATA
5919 024374  005204                                INC     R4                      ;
5920 024376  020204                                CMP     R2,R4                   ;IS R2 CORRECT
5921 024400  001401                                BEQ     7$                      ;YES GO ON
5922 024402  104001                                ERROR   +1              ;CPU ERROR
5923                                                                       ;NO GO TO ERROR
5924 024404  022142                     7$:        CMP     (R1)+,-(R2)             ;IS TEST LOCATION CORRECT
5925 024406  001401                                BEQ     8$                      ;YES GO ON
5926 024410  104001                                ERROR   +1              ;CPU ERROR
5927                                                                       ;NO GO TO ERROR
5928 024412  005202                     8$:        INC     R2                      ;POINT TO NEXT TEST LOCATION
5929 024414  005202                                INC     R2                      ;
5930 024416  021127  177777                        CMP     (R1),#177777            ;ARE WE DONE
5931 024422  001337                                BNE     1$                      ;NO GO TO 1$
5932 024424  012737  024470  000010                MOV     #T124E,@#10             ;SETUP NEW VECTOR
5933 024432  007302                                .WORD   7302                    ; TEST INSTRUCTION ILLEGAL MODE
5934 024434  104001                                ERROR   +1              ;CPU ERROR
5935                                                                       ;GO TO ERROR IF DIDN'T TRAP
5936 024436  000167  000032                        JMP     T124F
5937                                    ;
5938                                    ;
5939 024442  167604                     T124A:     .WORD   167604
5940 024444  000000                                .WORD   0
5941 024446  000001                                .WORD   1
```

```
********** DOUBLE OPERAND TESTS ******************

 5942 024450  177777                        .WORD   177777
 5943 024452  177777              T124B:    .WORD   177777
 5944 024454  177777                        .WORD   177777
 5945 024456  177777                        .WORD   177777
 5946 024460  030011              T124C:    .WORD   30011
 5947 024462  030004                        .WORD   30004
 5948 024464  030000                        .WORD   30000
 5949 024466  104001              T124D:    ERROR   +1              ;CPU ERROR
 5950                                                               ;GO TO ERROR IF TRAPPED
 5951 024470  005726              T124E:    TST     (SP)+                   ;CLEAN UP STACK
 5952 024472  005726                        TST     (SP)+                   ;
 5953 024474  012637  000010      T124F:    MOV     (SP)+,@#10              ;RESTORE VECTOR
 5954
 5955                             ;
 5958 024500                      TE125:
 5959                             ;       TEST MUL (MULTIPLY INST)
 5960 024500  005037  177776                CLR     @#177776                ;INIT PS
 5961 024504  012701  024740                MOV     #TE125A,R1              ;SETUP POINTERS TO TABLES
 5962
 5963 024510  010137  111462      1$:       MOV     R1,@#EXPDAT             ;
 5964 024514  062737  000002  111462        ADD     #2,@#EXPDAT             ;POINT TO SOURCE
 5965 024522  012703  122222                MOV     #122222,R3              ;INIT R3 TO A KNOWN STATE
 5966 024526  011102                        MOV     (R1),R2         ;INIT DESTINATION REG
 5967 024530  000277                        SCC                             ;SET ALL CC BITS
 5968 024532  070261  000002                MUL     2(R1),R2                ; TEST INSTRUCTION
 5969 024536  026137  000004  177776        CMP     4(R1),@#177776          ;IS PS CORRECT
 5970 024544  001401                        BEQ     2$                      ;YES GO ON
 5971 024546  104001                        ERROR   +1              ;CPU ERROR
 5972                                                               ;NO GO TO ERROR
 5973 024550  026103  000006      2$:       CMP     6(R1),R3                ;IS R3 CORRECT
 5974 024554  001401                        BEQ     3$                      ;YES GO ON
 5975 024556  104001                        ERROR   +1              ;CPU ERROR
 5976                                                               ;NO GO TO ERROR
 5977 024560  026102  000010      3$:       CMP     10(R1),R2               ;IS R2 CORRECT
 5978 024564  001401                        BEQ     4$                      ;YES GO ON
 5979 024566  104001                        ERROR   +1              ;CPU ERROR
 5980                                                               ;NO GO TO ERROR
 5981 024570  026177  000002  064664  4$:   CMP     2(R1),@EXPDAT           ;IS SOURCE LOCATION OK
 5982 024576  001401                        BEQ     5$                      ;YES GO ON
 5983 024600  104001                        ERROR   +1              ;CPU ERROR
 5984                                                               ;NO GO TO ERROR
 5985 024602  062701  000012      5$:       ADD     #12,R1                  ;GO TO NEXT TEST
 5986 024606  020127  025166                CMP     R1,#FIN125              ;ARE WE FINISHED
 5987 024612  001336                        BNE     1$                      ;NO GO TO 1$
 5988
 5989                             ;
 5990                             ;SECOND PART
 5991                             ;USING ODD REGISTER
 5992                             ;
 5993 024614  012701  024740      6$:       MOV     #TE125A,R1              ;SETUP POINTERS TO TABLES
 5994 024620                      7$:
 5995 024620  010102                        MOV     R1,R2                   ;
 5996 024622  012706  001000                MOV     #STBOT,R6               ;INIT R6 TO A KNOWN STATE
 5997 024626  012704  000004                MOV     #4,R4                   ;SETUP R4 VALUE
 5998 024632  011105                        MOV     (R1),R5         ;INIT DESTINATION REG
 5999 024634  000277                        SCC                             ;SET ALL CC BITS
 6000 024636  070561  000002                MUL     2(R1),R5                ; TEST INSTRUCTION
```

```
********** DOUBLE OPERAND TESTS ******************

 6001 024642  026137  000004  177776          CMP     4(R1),#177776          ;IS PS CORRECT
 6002 024650  001401                          BEQ     8$                     ;YES GO ON
 6003 024652  104001                          ERROR   .1                  ;CPU ERROR
 6004                                                                     ;NO GO TO ERROR
 6005 024654  026105  000006          8$:     CMP     6(R1),R5               ;IS R5 CORRECT
 6006 024660  001401                          BEQ     9$                     ;YES GO ON
 6007 024662  104001                          ERROR   .1                  ;CPU ERROR
 6008                                                                     ;NO GO TO ERROR
 6009 024664  020627  001000          9$:     CMP     R6,#STBOT              ;IS R6 CORRECT
 6010 024670  001403                          BEQ     10$                    ;YES GO ON
 6011 024672  012706  001000                  MOV     #STBOT,R6              ;RESTORE SP
 6012 024676  104001                          ERROR   .1                  ;CPU ERROR
 6013                                                                     ;NO GO TO ERROR
 6014 024700  005722                  10$:    TST     (R2)+                  ;POINT TO SOURCE OPERAND
 6015 024702  021261  000002                  CMP     (R2),2(R1)             ;IS SOURCE LOCATION OK
 6016 024706  001401                          BEQ     11$                    ;YES GO ON
 6017 024710  104001                          ERROR   .1                  ;CPU ERROR
 6018                                                                     ;NO GO TO ERROR
 6019 024712  020427  000004          11$:    CMP     R4,#4                  ;IS R4 CORRECT
 6020 024716  001401                          BEQ     12$                    ;YES GO ON
 6021 024720  104001                          ERROR   .1                  ;CPU ERROR
 6022                                                                     ;NO GO TO ERROR
 6023 024722  062701  000012          12$:    ADD     #12,R1                 ;
 6024 024726  020127  025166                  CMP     R1,#FIN125             ;ARE WE FINISHED
 6025 024732  001332                          BNE     7$                     ;NO GO TO 7$
 6026
 6027
 6028 024734  000167  000226                  JMP     FIN125
 6029                                 ;
 6030                                 ;
 6031 024740  177777                  TE125A: .WORD   177777                 ;MULTIPLICAND
 6032 024742  177777                          .WORD   177777                 ;MULTIPLIER
 6033 024744  000000                          .WORD   0
 6034 024746  000001                          .WORD   1
 6035 024750  000000                          .WORD   0
 6036
 6037 024752  006772                          .WORD   6772                   ;MULTIPLICAND
 6038 024754  100000                          .WORD   100000                 ;MULTIPLIER
 6039 024756  000011                          .WORD   11
 6040 024760  000000                          .WORD   0
 6041 024762  174403                          .WORD   174403
 6042
 6043 024764  177777                          .WORD   177777                 ;MULTIPLICAND
 6044 024766  077777                          .WORD   77777                  ;MULTIPLIER
 6045 024770  000010                          .WORD   10
 6046 024772  100001                          .WORD   100001
 6047 024774  177777                          .WORD   177777
 6048
 6049 024776  077777                          .WORD   77777                  ;MULTIPLICAND
 6050 025000  000456                          .WORD   456                    ;MULTIPLIER
 6051 025002  000001                          .WORD   1
 6052 025004  177322                          .WORD   177322
 6053 025006  000226                          .WORD   226
 6054
 6055 025010  173210                          .WORD   173210                 ;MULTIPLICAND
 6056 025012  000000                          .WORD   0                      ;MULTIPLIER
 6057 025014  000004                          .WORD   4
```

********** DOUBLE OPERAND TESTS **********

```
6058 025016  000000                    .WORD   0
6059 025020  000000                    .WORD   0
6060
6061 025022  000000                    .WORD   0               ;MULTIPLICAND
6062 025024  003251                    .WORD   3251            ;MULTIPLIER
6063 025026  000004                    .WORD   4
6064 025030  000000                    .WORD   0
6065 025032  000000                    .WORD   0
6066
6067 025034  000000                    .WORD   0               ;MULTIPLICAND
6068 025036  000000                    .WORD   0               ;MULTIPLIER
6069 025040  000004                    .WORD   4
6070 025042  000000                    .WORD   0
6071 025044  000000                    .WORD   0
6072
6073 025046  100000                    .WORD   100000          ;MULTIPLICAND
6074 025050  000001                    .WORD   1               ;MULTIPLIER
6075 025052  000010                    .WORD   10
6076 025054  100000                    .WORD   100000
6077 025056  177777                    .WORD   177777
6078
6079 025060  077777                    .WORD   77777           ;MULTIPLICAND
6080 025062  000001                    .WORD   1               ;MULTIPLIER
6081 025064  000000                    .WORD   0
6082 025066  077777                    .WORD   77777
6083 025070  000000                    .WORD   0
6084
6085 025072  000010                    .WORD   10              ;MULTIPLICAND
6086 025074  010000                    .WORD   10000           ;MULTIPLIER
6087 025076  000001                    .WORD   1
6088 025100  100000                    .WORD   100000
6089 025102  000000                    .WORD   0
6090
6091 025104  001452                    .WORD   1452            ;MULTIPLICAND
6092 025106  034527                    .WORD   34527           ;MULTIPLIER
6093 025110  000001                    .WORD   1
6094 025112  066506                    .WORD   66506
6095 025114  000265                    .WORD   265
6096
6097 025116  000007                    .WORD   7               ;MULTIPLICAND
6098 025120  000400                    .WORD   400             ;MULTIPLIER
6099 025122  000000                    .WORD   0
6100 025124  003400                    .WORD   3400
6101 025126  000000                    .WORD   0
6102
6103 025130  000002                    .WORD   2               ;MULTIPLICAND
6104 025132  100000                    .WORD   100000          ;MULTIPLIER
6105 025134  000011                    .WORD   11
6106 025136  000000                    .WORD   0
6107 025140  177777                    .WORD   177777
6108
6109 025142  100000                    .WORD   100000          ;MULTIPLICAND
6110 025144  077777                    .WORD   77777           ;MULTIPLIER
6111 025146  000011                    .WORD   11
6112 025150  100000                    .WORD   100000
6113 025152  140000                    .WORD   140000
6114
```

```
********** DOUBLE OPERAND TESTS ******************

 6115 025154  000001                          .WORD   1                       ;MULTIPLICAND
 6116 025156  177777                          .WORD   177777                  ;MULTIPLIER
 6117 025160  000010                          .WORD   10
 6118 025162  177777                          .WORD   177777
 6119 025164  177777                          .WORD   177777
 6120 025166                          FIN125:
 6121                                 ;
 6124 025166                          TE126:
 6125                                 ;       TEST DIV (DIVIDE INST)
 6126 025166  005037  177776                  CLR     @#177776                ;INIT PSW
 6127 025172  005006                          CLR     R6                      ;INIT SP
 6128 025174  013705  000000                  MOV     @#0,R5                  ;SAVE VECTORS
 6129 025200  013701  000002                  MOV     @#2,R1                  ;
 6130 025204  012737  000137  000000          MOV     #137,@#0                ;SETUP NEW VECTORS
 6131 025212  012737  025234  000002          MOV     #TE126A,@#2             ;
 6132 025220  000277                          SCC                             ;SET ALL CC BITS
 6133 025222  071627  000002                  DIV     #2,R6                   ; TEST INSTRUCTION
 6134 025226  012706  001000          A126:   MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
 6135 025232  104001                          ERROR   +1                      ;CPU ERROR
 6136                                                                         ;IF R7 ISN'T CORRECT GO TO ERROR
 6137 025234  022737  000000  177776  TE126A: CMP     #0,@#177776             ;IS PS CORRECT
 6138 025242  001403                          BEQ     1$                      ;YES GO ON
 6139 025244  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
 6140 025250  104001                          ERROR   +1                      ;CPU ERROR
 6141                                                                         ;NO GO TO ERROR
 6142 025252  012704  025226          1$:     MOV     #A126,R4                ;SETUP EXPECTED DATA
 6143 025256  006204                          ASR     R4                      ;
 6144 025260  020406                          CMP     R4,R6                   ;IS R6 CORRECT
 6145 025262  001403                          BEQ     2$                      ;YES GO ON
 6146 025264  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
 6147 025270  104001                          ERROR   +1                      ;CPU ERROR
 6148                                                                         ;NO GO TO ERROR
 6149 025272  010537  000000          2$:     MOV     R5,@#0                  ;RESTORE VECTORS
 6150 025276  010137  000002                  MOV     R1,@#2                  ;
 6151 025302  012706  001000                  MOV     #STBOT,R6               ;INIT SP
 6152 025306  012702  000006                  MOV     #6,R2                   ;INIT GPR 2
 6153 025312  012703  000047                  MOV     #47,R3                  ;INIT GPR 3
 6154 025316  000277                          SCC                             ;SET ALL CC BITS
 6155 025320  071302                          DIV     R2,R3                   ; TEST INSTRUCTION
 6156 025322  022737  000002  177776          CMP     #2,@#177776             ;IS PS CORRECT
 6157 025330  001401                          BEQ     3$                      ;YES GO ON
 6158 025332  104001                          ERROR   +1                      ;CPU ERROR
 6159                                                                         ;NO GO TO ERROR
 6160 025334  022702  000006          3$:     CMP     #6,R2                   ;IS R2 CORRECT
 6161 025340  001401                          BEQ     4$                      ;YES GO ON
 6162 025342  104001                          ERROR   +1                      ;CPU ERROR
 6163                                                                         ;NO GO TO ERROR
 6164 025344  022703  000047          4$:     CMP     #47,R3                  ;IS R3 CORRECT
 6165 025350  001401                          BEQ     5$                      ;YES GO ON
 6166 025352  104001                          ERROR   +1                      ;CPU ERROR
 6167                                                                         ;NO GO TO ERROR
 6168 025354  005004                  5$:     CLR     R4                      ;INIT R4
 6169 025356  012705  000004                  MOV     #4,R5                   ;INIT R5
 6170 025362  000277                          SCC                             ;SET ALL CC BITS
 6171 025364  071427  000000                  DIV     #0,R4                   ; TEST INSTRUCTION
 6172 025370  022737  000007  177776          CMP     #7,@#177776             ;IS PS CORRECT
 6173 025376  001401                          BEQ     6$                      ;YES GO ON
```

```
025400  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025402  022704  000000          6$:     CMP     #0,R4                   ;IS R4 CORRECT
025406  001401                          BEQ     7$                      ;YES GO ON
025410  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025412  022705  000004          7$:     CMP     #4,R5                   ;IS R5 CORRECT
025416  001401                          BEQ     8$                      ;YES GO ON
025420  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025422  012700  000004          8$:     MOV     #4,R0                   ;INIT R0
025426  012705  000010                  MOV     #10,R5                  ;INIT R5
025432  005004                          CLR     R4                      ;INIT R4
025434  000277                          SCC                             ;SET ALL CC BITS
025436  071400                          DIV     R0,R4                   ; TEST INSTRUCTION
025440  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
025446  001405                          BEQ     9$                      ;YES GO ON
025450  010067  152724                  MOV     R0,400                  ;SAVE R0
025454  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025456  016700  152716                  MOV     400,R0                  ;RESTORE R0
025462  022700  000004          9$:     CMP     #4,R0                   ;IS R0 CORRECT
025466  001401                          BEQ     10$                     ;YES GO ON
025470  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025472  022704  000002          10$:    CMP     #2,R4                   ;IS R4 CORRECT
025476  001401                          BEQ     11$                     ;YES GO ON
025500  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025502  022705  000000          11$:    CMP     #0,R5                   ;IS R5 CORRECT
025506  001401                          BEQ     12$                     ;YES GO ON
025510  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025512  012705  000010          12$:    MOV     #10,R5                  ;INIT R5
025516  005004                          CLR     R4                      ;INIT R4
025520  000277                          SCC                             ;SET ALL CC BITS
025522  071427  000003                  DIV     #3,R4                   ; TEST INSTRUCTION
025526  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
025534  001401                          BEQ     13$                     ;YES GO ON
025536  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025540  022704  000002          13$:    CMP     #2,R4                   ;IS R4 CORRECT
025544  001401                          BEQ     14$                     ;YES GO ON
025546  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
025550  022705  000002          14$:    CMP     #2,R5                   ;IS R5 CORRECT
025554  001401                          BEQ     15$                     ;YES GO ON
025556  104001                          ERROR   +1              ;CPU ERROR
                                                                ;NO GO TO ERROR
                                ;
                                ;
                                ;
025560  012701  025710          15$:    MOV     #TE126B,R1              ;SETUP POINTERS TO TABLES

025564  010137  111462          16$:    MOV     R1,@#EXPDAT             ;SAVE A COPY OF R1
025570  011104                          MOV     (R1),R4         ;INIT R4
025572  016103  000004                  MOV     4(R1),R3                ;SAVE SOURCE
```

********** DOUBLE OPERAND TESTS **********

```
6231                                            ;
6232 025576  016105  000002                     MOV     2(R1),R5                ;INIT R5
6233 025602  000277                             SCC                             ;SET ALL CC BITS
6234 025604  071461  000004                     DIV     4(R1),R4                ; TEST INSTRUCTION
6235 025610  026137  000006  177776             CMP     6(R1),@#177776          ;IS PS CORRECT
6236 025616  001401                             BEQ     17$                     ;YES GO ON
6237 025620  104001                             ERROR   .+1             ;CPU ERROR
6238                                                                    ;NO GO TO ERROR
6239 025622  026105  000010             17$:    CMP     10(R1),R5               ;IS R5 CORRECT
6240 025626  001401                             BEQ     18$                     ;YES GO ON
6241 025630  104001                             ERROR   .+1             ;CPU ERROR
6242                                                                    ;NO GO TO ERROR
6243 025632  026104  000012             18$:    CMP     12(R1),R4               ;IS R4 CORRECT
6244 025636  001401                             BEQ     19$                     ;YES GO ON
6245 025640  104001                             ERROR   .+1             ;CPU ERROR
6246                                                                    ;NO GO TO ERROR
6247 025642  023701  111462             19$:    CMP     @#EXPDAT,R1             ;IS R1 CORRECT
6248 025646  001403                             BEQ     20$                     ;YES GO ON
6249 025650  104001                             ERROR   .+1             ;CPU ERROR
6250                                                                    ;NO GO TO ERROR
6251 025652  013701  111462                     MOV     @#EXPDAT,R1             ;RESTORE CORRECT VALUE
6252 025656  026103  000004             20$:    CMP     4(R1),R3                ;IS SOURCE CORRECT
6253 025662  001403                             BEQ     21$                     ;YES GO ON
6254 025664  104001                             ERROR   .+1             ;CPU ERROR
6255                                                                    ;NO GO TO ERROR
6256 025666  010361  000004                     MOV     R3,4(R1)                ;TRY TO RESTORE CODE
6257 025672  062701  000014             21$:    ADD     #14,R1                  ;POINT TO NEXT LOCATION
6258 025676  021127  000333                     CMP     (R1),#333               ;ARE WE DONE
6259 025702  001330                             BNE     16$                     ;NO GO TO 16$
6260
6261
6262 025704  000167  000316                     JMP     FIN126
6263                                        ;
6264                                        ;
6265 025710  177777                     TE1268: .WORD   177777  ;DIVIDEND
6266 025712  177777                             .WORD   177777  ;INIT R5
6267 025714  177777                             .WORD   177777  ;DIVISOR
6268 025716  000000                             .WORD   0       ;PSW
6269 025720  000000                             .WORD   0       ;R5 RESULT
6270 025722  000001                             .WORD   1       ;R4 RESULT
6271
6272 025724  000000                             .WORD   0       ;DIVIDEND
6273 025726  177777                             .WORD   177777  ;INIT R5
6274 025730  177777                             .WORD   177777  ;DIVISOR
6275 025732  000012                             .WORD   12      ;PSW
6276 025734  177777                             .WORD   177777  ;R5 RESULT
6277 025736  000000                             .WORD   0       ;R4 RESULT
6278
6279 025740  177777                             .WORD   177777  ;DIVIDEND
6280 025742  000000                             .WORD   0       ;INIT R5
6281 025744  177777                             .WORD   177777  ;DIVISOR
6282 025746  000002                             .WORD   2       ;PSW
6283 025750  000000                             .WORD   0       ;R5 RESULT
6284 025752  177777                             .WORD   177777  ;R4 RESULT
6285
6286 025754  000000                             .WORD   0       ;DIVIDEND
6287 025756  007642                             .WORD   7642    ;INIT R5
```

********** DOUBLE OPERAND TESTS **********

```
6288 025760  007643                    .WORD   7643    ;DIVISOR
6289 025762  000004                    .WORD   4       ;PSW
6290 025764  007642                    .WORD   7642    ;R5 RESULT
6291 025766  000000                    .WORD   0       ;R4 RESULT
6292
6293 025770  000000                    .WORD   0       ;DIVIDEND
6294 025772  000137                    .WORD   137     ;INIT R5
6295 025774  177543                    .WORD   177543  ;DIVISOR
6296 025776  000004                    .WORD   4       ;PSW
6297 026000  000137                    .WORD   137     ;R5 RESULT
6298 026002  000000                    .WORD   0       ;R4 RESULT
6299
6300 026004  000000                    .WORD   0       ;DIVIDEND
6301 026006  007643                    .WORD   7643    ;INIT R5
6302 026010  007643                    .WORD   7643    ;DIVISOR
6303 026012  000000                    .WORD   0       ;PSW
6304 026014  000000                    .WORD   0       ;R5 RESULT
6305 026016  000001                    .WORD   1       ;R4 RESULT
6306
6307 026020  100000                    .WORD   100000  ;DIVIDEND
6308 026022  004376                    .WORD   4376    ;INIT R5
6309 026024  010021                    .WORD   10021   ;DIVISOR
6310 026026  000012                    .WORD   12      ;PSW
6311 026030  004376                    .WORD   4376    ;R5 RESULT
6312 026032  100000                    .WORD   100000  ;R4 RESULT
6313
6314 026034  177700                    .WORD   177700  ;DIVIDEND
6315 026036  170033                    .WORD   170033  ;INIT R5
6316 026040  010021                    .WORD   10021   ;DIVISOR
6317 026042  000010                    .WORD   10      ;PSW
6318 026044  171307                    .WORD   171307  ;R5 RESULT
6319 026046  176024                    .WORD   176024  ;R4 RESULT
6320
6321 026050  177700                    .WORD   177700  ;DIVIDEND
6322 026052  170033                    .WORD   170033  ;INIT R5
6323 026054  167757                    .WORD   167757  ;DIVISOR
6324 026056  000000                    .WORD   0       ;PSW
6325 026060  171307                    .WORD   171307  ;R5 RESULT
6326 026062  001754                    .WORD   1754    ;R4 RESULT
6327
6328 026064  000000                    .WORD   0       ;DIVIDEND
6329 026066  177777                    .WORD   177777  ;INIT R5
6330 026070  000001                    .WORD   1       ;DIVISOR
6331 026072  000002                    .WORD   2       ;PSW
6332 026074  177777                    .WORD   177777  ;R5 RESULT
6333 026076  000000                    .WORD   0       ;R4 RESULT
6334
6335 026100  177777                    .WORD   177777  ;DIVIDEND
6336 026102  045716                    .WORD   45716   ;INIT R5
6337 026104  000001                    .WORD   1       ;DIVISOR
6338 026106  000012                    .WORD   12      ;PSW
6339 026110  045716                    .WORD   45716   ;R5 RESULT
6340 026112  177777                    .WORD   177777  ;R4 RESULT
6341
6342 026114  000000                    .WORD   0       ;DIVIDEND
6343 026116  000002                    .WORD   2       ;INIT R5
6344 026120  177770                    .WORD   177770  ;DIVISOR
```

```
********** DOUBLE OPERAND TESTS **********

6345 026122  000004                        .WORD   4       ;PSW
6346 026124  000002                        .WORD   2       ;R5 RESULT
6347 026126  000000                        .WORD   0       ;R4 RESULT
6348
6349 026130  177777                        .WORD   177777  ;DIVIDEND
6350 026132  177776                        .WORD   177776  ;INIT R5
6351 026134  000010                        .WORD   10      ;DIVISOR
6352 026136  000004                        .WORD   4       ;PSW
6353 026140  177776                        .WORD   177776  ;R5 RESULT
6354 026142  000000                        .WORD   0       ;R4 RESULT
6355
6356 026144  000001                        .WORD   1       ;DIVIDEND
6357 026146  177777                        .WORD   177777  ;INIT R5
6358 026150  000001                        .WORD   1       ;DIVISOR
6359 026152  000002                        .WORD   2       ;PSW
6360 026154  177777                        .WORD   177777  ;R5 RESULT
6361 026156  000001                        .WORD   1       ;R4 RESULT
6362
6363 026160  000001                        .WORD   1       ;DIVIDEND
6364 026162  000000                        .WORD   0       ;INIT R5
6365 026164  000002                        .WORD   2       ;DIVISOR
6366 026166  000002                        .WORD   2       ;PSW
6367 026170  000000                        .WORD   0       ;R5 RESULT
6368 026172  000001                        .WORD   1       ;R4 RESULT
6369
6370 026174  000001                        .WORD   1       ;DIVIDEND
6371 026176  000000                        .WORD   0       ;INIT R5
6372 026200  000003                        .WORD   3       ;DIVISOR
6373 026202  000000                        .WORD   0       ;PSW
6374 026204  000001                        .WORD   1       ;R5 RESULT
6375 026206  052525                        .WORD   52525   ;R4 RESULT
6376
6377 026210  000023                        .WORD   23      ;DIVIDEND
6378 026212  016054                        .WORD   16054   ;INIT R5
6379 026214  016537                        .WORD   16537   ;DIVISOR
6380 026216  000000                        .WORD   0       ;PSW
6381 026220  010222                        .WORD   10222   ;R5 RESULT
6382 026222  000246                        .WORD   246     ;R4 RESULT
6383
6384 026224  000333                        .WORD   333
6385 026226                        FIN126:
6386                               ;
6389 026226                        TE127:
6390                               ;       TEST ASH (ARITHMETIC SHIFT)
6391 026226  005037  177776                CLR     @#177776                ;INIT PSW
6392 026232  012702  000001                MOV     #1,R2                   ;SETUP OPERAND
6393 026236  000277                        SCC                             ;SET ALL CC BITS
6394 026240  072202                        ASH     R2,R2                   ; TEST INSTRUCTION
6395 026242  022737  000000  177776        CMP     #0,@#177776             ;IS PS CORRECT
6396 026250  001401                        BEQ     1$                      ;YES GO ON
6397 026252  104001                        ERROR   .1              ;CPU ERROR
6398                                                               ;NO GO TO ERROR
6399 026254  020227  000002          1$:   CMP     R2,#2                   ;IS R2 CORRECT
6400 026260  001401                        BEQ     2$                      ;YES GO ON
6401 026262  104001                        ERROR   .1              ;CPU ERROR
6402                                                               ;NO GO TO ERROR
6403 026264  012702  100000          2$:   MOV     #100000,R2              ;SETUP R2
```

```
026270  012703  000001                  MOV     #1,R3                   ;SETUP R3
026274  000257                          CCC                             ;CLEAR ALL CC BITS
026276  072203                          ASH     R3,R2                   ; TEST INSTRUCTION
026300  022737  000007  177776          CMP     #7,@#177776             ;IS PS CORRECT
026306  001401                          BEQ     3$                      ;YES GO ON
026310  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026312  020327  000001          3$:     CMP     R3,#1                   ;IS R3 CORRECT
026316  001401                          BEQ     4$                      ;YES GO ON
026320  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026322  020227  000000          4$:     CMP     R2,#0                   ;IS R2 CORRECT
026326  001401                          BEQ     5$                      ;YES GO ON
026330  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026332  012701  026426          5$:     MOV     #TE127A,R1              ;SETUP POINTERS TO TABLES

026336  010103                  6$:     MOV     R1,R3                   ;
026340  016102  000002                  MOV     2(R1),R2                ;SETUP R2
026344  000277                          SCC                             ;SET ALL CC BITS
026346  072211                          ASH     (R1),R2                 ; TEST INSTRUCTION
026350  026137  000004  177776          CMP     4(R1),@#177776          ;IS PS CORRECT
026356  001401                          BEQ     7$                      ;YES GO ON
026360  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026362  026102  000006          7$:     CMP     6(R1),R2                ;IS R2 CORRECT
026366  001401                          BEQ     8$                      ;YES GO ON
026370  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026372  020301                  8$:     CMP     R3,R1                   ;IS R1 CORRECT
026374  001402                          BEQ     9$                      ;YES GO ON
026376  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
026400  010301                          MOV     R3,R1                   ;RESTORE R1
026402  021311                  9$:     CMP     (R3),(R1)               ;IS SOURCE CORRECT
026404  001401                          BEQ     10$                     ;YES GO ON
026406  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;NO GO TO ERROR
                                                                          ;SOURCE LOOKS INCORRECT
026410  062701  000010          10$:    ADD     #10,R1                  ;INCREMENT POINTER
026414  020127  026666                  CMP     R1,#FIN127              ;ARE WE DONE
026420  001346                          BNE     6$                      ;NO GO TO 6$


026422  000167  000240                  JMP     FIN127
                                ;
                                ;
026426  177761                  TE127A: .WORD   177761            ;SOURCE
026430  077777                          .WORD   77777             ;DEST
026432  000005                          .WORD   5
026434  000000                          .WORD   0

026436  177700                          .WORD   177700            ;SOURCE
026440  017777                          .WORD   17777             ;DEST
026442  000000                          .WORD   0
026444  017777                          .WORD   17777
```

********** DOUBLE OPERAND TESTS **********

```
6461 026446  177700                    .WORD   177700          ;SOURCE
6462 026450  100000                    .WORD   100000          ;DEST
6463 026452  000010                    .WORD   10
6464 026454  100000                    .WORD   100000
6465
6466 026456  177777                    .WORD   177777          ;SOURCE
6467 026460  100000                    .WORD   100000          ;DEST
6468 026462  000010                    .WORD   10
6469 026464  140000                    .WORD   140000
6470
6471 026466  177737                    .WORD   177737          ;SOURCE
6472 026470  177777                    .WORD   177777          ;DEST
6473 026472  000011                    .WORD   11
6474 026474  177777                    .WORD   177777
6475
6476 026476  177706                    .WORD   177706          ;SOURCE
6477 026500  102000                    .WORD   102000          ;DEST
6478 026502  000007                    .WORD   7
6479 026504  000000                    .WORD   0
6480
6481 026506  177710                    .WORD   177710          ;SOURCE
6482 026510  017777                    .WORD   17777           ;DEST
6483 026512  000013                    .WORD   13
6484 026514  177400                    .WORD   177400
6485
6486 026516  177713                    .WORD   177713          ;SOURCE
6487 026520  000012                    .WORD   12              ;DEST
6488 026522  000000                    .WORD   0
6489 026524  050000                    .WORD   50000
6490
6491 026526  177707                    .WORD   177707          ;SOURCE
6492 026530  170001                    .WORD   170001          ;DEST
6493 026532  000002                    .WORD   2
6494 026534  000200                    .WORD   200
6495
6496 026536  177717                    .WORD   177717          ;SOURCE
6497 026540  000001                    .WORD   1               ;DEST
6498 026542  000012                    .WORD   12
6499 026544  100000                    .WORD   100000
6500
6501 026546  177740                    .WORD   177740          ;SOURCE
6502 026550  017777                    .WORD   17777           ;DEST
6503 026552  000004                    .WORD   4
6504 026554  000000                    .WORD   0
6505
6506 026556  177771                    .WORD   177771          ;SOURCE
6507 026560  150000                    .WORD   150000          ;DEST
6508 026562  000010                    .WORD   10
6509 026564  177640                    .WORD   177640
6510
6511 026566  177742                    .WORD   177742          ;SOURCE
6512 026570  100000                    .WORD   100000          ;DEST
6513 026572  000011                    .WORD   11
6514 026574  177777                    .WORD   177777
6515
6516 026576  177764                    .WORD   177764          ;SOURCE
6517 026600  100000                    .WORD   100000          ;DEST
```

```
********** DOUBLE OPERAND TESTS ******************

 6518 026602  000010                          .WORD   10
 6519 026604  177770                          .WORD   177770
 6520
 6521 026606  177750                          .WORD   177750          ;SOURCE
 6522 026610  052525                          .WORD   52525           ;DEST
 6523 026612  000004                          .WORD   4
 6524 026614  000000                          .WORD   0
 6525
 6526 026616  177760                          .WORD   177760          ;SOURCE
 6527 026620  100000                          .WORD   100000          ;DEST
 6528 026622  000011                          .WORD   11
 6529 026624  177777                          .WORD   177777
 6530
 6531 026626  177770                          .WORD   177770          ;SOURCE
 6532 026630  100000                          .WORD   100000          ;DEST
 6533 026632  000010                          .WORD   10
 6534 026634  177600                          .WORD   177600
 6535
 6536 026636  177712                          .WORD   177712          ;SOURCE
 6537 026640  004367                          .WORD   4367            ;DEST
 6538 026642  000013                          .WORD   13
 6539 026644  156000                          .WORD   156000
 6540
 6541 026646  177764                          .WORD   177764          ;SOURCE
 6542 026650  017777                          .WORD   17777           ;DEST
 6543 026652  000001                          .WORD   1
 6544 026654  000001                          .WORD   1
 6545
 6546 026656  177701                          .WORD   177701          ;SOURCE
 6547 026660  110000                          .WORD   110000          ;DEST
 6548 026662  000003                          .WORD   3
 6549 026664  020000                          .WORD   20000
 6550
 6551 026666  000240                  FIN127: NOP
 6552                                 ;
 6555 026670                          TE130:
 6556                                 ;       TEST ASHC (ARITHMETIC SHIFT COMBINED)
 6557 026670  005037  177776                  CLR     @#177776                ;INIT PSW
 6558 026674  012701  000023                  MOV     #23,R1                  ;SETUP R1
 6559 026700  012705  052525                  MOV     #52525,R5               ;SETUP R5
 6560 026704  005004                          CLR     R4                      ;SETUP R4
 6561 026706  000277                          SCC                             ;SET ALL CC BITS
 6562 026710  073401                          ASHC    R1,R4                   ; TEST INSTRUCTION
 6563 026712  023727  177776  000012          CMP     @#177776,#12            ;IS PS CORRECT
 6564 026720  001401                          BEQ     1$                      ;YES GO ON
 6565 026722  104001                          ERROR   +1              ;CPU ERROR
 6566                                                                 ;NO GO TO ERROR
 6567 026724  020127  000023          1$:     CMP     R1,#23                  ;IS R1 CORRECT
 6568 026730  001401                          BEQ     2$                      ;YES GO ON
 6569 026732  104001                          ERROR   +1              ;CPU ERROR
 6570                                                                 ;NO GO TO ERROR
 6571 026734  020427  125250          2$:     CMP     R4,#125250              ;IS R4 CORRECT
 6572 026740  001401                          BEQ     3$                      ;YES GO ON
 6573 026742  104001                          ERROR   +1              ;CPU ERROR
 6574                                                                 ;NO GO TO ERROR
 6575 026744  020527  000000          3$:     CMP     R5,#0                   ;IS R5 CORRECT
 6576 026750  001401                          BEQ     4$                      ;YES GO ON
```

```
********** DOUBLE OPERAND TESTS ******************

6577 026752  104001                          ERROR   ;1              ;CPU ERROR
6578                                                                 ;NO GO TO ERROR
6579 026754  012703  052525          4$:     MOV     #52525,R3       ;SETUP R3
6580 026760  005002                          CLR     R2              ;SETUP R2
6581 026762  012704  164731                  MOV     #164731,R4      ;SETUP R4
6582 026766  000277                          SCC                     ;SET ALL CC BITS
6583 026770  073327  000023                  ASHC    #23,R3          ; TEST INSTRUCTION
6584 026774  023727  177776  000012          CMP     @#177776,#12    ;IS PS CORRECT
6585 027002  001401                          BEQ     5$              ;YES GO ON
6586 027004  104001                          ERROR   ;1              ;CPU ERROR
6587                                                                 ;NO GO TO ERROR
6588 027006  020227  000000          5$:     CMP     R2,#0           ;IS R2 CORRECT
6589 027012  001401                          BEQ     6$              ;YES GO ON
6590 027014  104001                          ERROR   ;1              ;CPU ERROR
6591                                                                 ;NO GO TO ERROR
6592 027016  020327  000000          6$:     CMP     R3,#0           ;IS R3 CORRECT
6593 027022  001401                          BEQ     7$              ;YES GO ON
6594 027024  104001                          ERROR   ;1              ;CPU ERROR
6595                                                                 ;NO GO TO ERROR
6596 027026  020427  164731          7$:     CMP     R4,#164731      ;IS R4 CORRECT
6597 027032  001401                          BEQ     8$              ;YES GO ON
6598 027034  104001                          ERROR   ;1              ;CPU ERROR
6599                                                                 ;NO GO TO ERROR
6600                                                                 ;
6601                                                                 ;
6602 027036  012701  027146          8$:     MOV     #TE130A,R1      ;SETUP POINTERS TO TABLES
6603
6604 027042  010104                  9$:     MOV     R1,R4           ;SAVE A COPY OF R1
6605 027044  016102  000002                  MOV     2(R1),R2        ;SETUP R2
6606 027050  016103  000004                  MOV     4(R1),R3        ;SETUP R3
6607 027054  000277                          SCC                     ;SET ALL CC BITS
6608 027056  073211                          ASHC    (R1),R2         ; TEST INSTRUCTION
6609 027060  023761  177776  000006          CMP     @#177776,6(R1)  ;IS PS CORRECT
6610 027066  001401                          BEQ     10$             ;YES GO ON
6611 027070  104001                          ERROR   ;1              ;CPU ERROR
6612                                                                 ;NO GO TO ERROR
6613 027072  026102  000010          10$:    CMP     10(R1),R2       ;IS R2 CORRECT
6614 027076  001401                          BEQ     11$             ;YES GO ON
6615 027100  104001                          ERROR   ;1              ;CPU ERROR
6616                                                                 ;NO GO TO ERROR
6617 027102  026103  000012          11$:    CMP     12(R1),R3       ;IS R3 CORRECT
6618 027106  001401                          BEQ     12$             ;YES GO ON
6619 027110  104001                          ERROR   ;1              ;CPU ERROR
6620                                                                 ;NO GO TO ERROR
6621 027112  020401                  12$:    CMP     R4,R1           ;IS R1 CORRECT
6622 027114  001402                          BEQ     13$             ;YES GO ON
6623 027116  104001                          ERROR   ;1              ;CPU ERROR
6624                                                                 ;NO GO TO ERROR
6625 027120  010401                          MOV     R4,R1
6626 027122  021114                  13$:    CMP     (R1),(R4)       ;IS SOURCE CORRECT
6627 027124  001401                          BEQ     14$             ;YES GO ON
6628 027126  104001                          ERROR   ;1              ;CPU ERROR
6629                                                                 ;NO GO TO ERROR
6630                                                                 ;POSSIBLE SOURCE CODE CORRUPTION
6631 027130  062701  000014          14$:    ADD     #14,R1          ;GO TO NEXT TEST
6632 027134  020127  027556                  CMP     R1,#FIN130      ;ARE WE DONE
6633 027140  001340                          BNE     9$              ;NO GO TO 9$
```

********** DOUBLE OPERAND TESTS **********

```
6634
6635
6636 027142  000167  000410                    JMP     FIN130
6637                                ;
6638                                ;
6639 027146  177700                TE130A: .WORD   177700          ;SOURCE
6640 027150  100125                        .WORD   100125          ;DESTINATION WORD 1
6641 027152  177777                        .WORD   177777          ;DESTINATION WORD 2
6642 027154  000010                        .WORD   10              ;TEST PSW
6643 027156  100125                        .WORD   100125          ;RESULT WORD 1
6644 027160  177777                        .WORD   177777          ;RESULT WORD 2
6645
6646 027162  177777                        .WORD   177777          ;SOURCE
6647 027164  000001                        .WORD   1               ;DESTINATION WORD 1
6648 027166  000000                        .WORD   0               ;DESTINATION WORD 2
6649 027170  000000                        .WORD   0               ;TEST PSW
6650 027172  000000                        .WORD   0               ;RESULT WORD 1
6651 027174  100000                        .WORD   100000          ;RESULT WORD 2
6652
6653 027176  177701                        .WORD   177701          ;SOURCE
6654 027200  047777                        .WORD   47777           ;DESTINATION WORD 1
6655 027202  100000                        .WORD   100000          ;DESTINATION WORD 2
6656 027204  000012                        .WORD   12              ;TEST PSW
6657 027206  117777                        .WORD   117777          ;RESULT WORD 1
6658 027210  000000                        .WORD   0               ;RESULT WORD 2
6659
6660 027212  177706                        .WORD   177706          ;SOURCE
6661 027214  004256                        .WORD   4256            ;DESTINATION WORD 1
6662 027216  177700                        .WORD   177700          ;DESTINATION WORD 2
6663 027220  000002                        .WORD   2               ;TEST PSW
6664 027222  025677                        .WORD   25677           ;RESULT WORD 1
6665 027224  170000                        .WORD   170000          ;RESULT WORD 2
6666
6667 027226  177711                        .WORD   177711          ;SOURCE
6668 027230  065700                        .WORD   65700           ;DESTINATION WORD 1
6669 027232  000012                        .WORD   12              ;DESTINATION WORD 2
6670 027234  000013                        .WORD   13              ;TEST PSW
6671 027236  100000                        .WORD   100000          ;RESULT WORD 1
6672 027240  012000                        .WORD   12000           ;RESULT WORD 2
6673
6674 027242  177737                        .WORD   177737          ;SOURCE
6675 027244  000000                        .WORD   0               ;DESTINATION WORD 1
6676 027246  000001                        .WORD   1               ;DESTINATION WORD 2
6677 027250  000004                        .WORD   4               ;TEST PSW
6678 027252  000000                        .WORD   0               ;RESULT WORD 1
6679 027254  000000                        .WORD   0               ;RESULT WORD 2
6680
6681 027256  177736                        .WORD   177736          ;SOURCE
6682 027260  000000                        .WORD   0               ;DESTINATION WORD 1
6683 027262  000001                        .WORD   1               ;DESTINATION WORD 2
6684 027264  000000                        .WORD   0               ;TEST PSW
6685 027266  040000                        .WORD   40000           ;RESULT WORD 1
6686 027270  000000                        .WORD   0               ;RESULT WORD 2
6687
6688 027272  177740                        .WORD   177740          ;SOURCE
6689 027274  100000                        .WORD   100000          ;DESTINATION WORD 1
6690 027276  000000                        .WORD   0               ;DESTINATION WORD 2
```

********** DOUBLE OPERAND TESTS **********

```
6691 027300  000011                    .WORD   11              ;TEST PSW
6692 027302  177777                    .WORD   177777          ;RESULT WORD 1
6693 027304  177777                    .WORD   177777          ;RESULT WORD 2
6694
6695 027306  177725                    .WORD   177725          ;SOURCE
6696 027310  177777                    .WORD   177777          ;DESTINATION WORD 1
6697 027312  174000                    .WORD   174000          ;DESTINATION WORD 2
6698 027314  000007                    .WORD   7               ;TEST PSW
6699 027316  000000                    .WORD   0               ;RESULT WORD 1
6700 027320  000000                    .WORD   0               ;RESULT WORD 2
6701
6702 027322  177724                    .WORD   177724          ;SOURCE
6703 027324  177777                    .WORD   177777          ;DESTINATION WORD 1
6704 027326  174000                    .WORD   174000          ;DESTINATION WORD 2
6705 027330  000011                    .WORD   11              ;TEST PSW
6706 027332  100000                    .WORD   100000          ;RESULT WORD 1
6707 027334  000000                    .WORD   0               ;RESULT WORD 2
6708
6709 027336  177733                    .WORD   177733          ;SOURCE
6710 027340  177777                    .WORD   177777          ;DESTINATION WORD 1
6711 027342  157023                    .WORD   157023          ;DESTINATION WORD 2
6712 027344  000012                    .WORD   12              ;TEST PSW
6713 027346  114000                    .WORD   114000          ;RESULT WORD 1
6714 027350  000000                    .WORD   0               ;RESULT WORD 2
6715
6716 027352  177727                    .WORD   177727          ;SOURCE
6717 027354  000000                    .WORD   0               ;DESTINATION WORD 1
6718 027356  177777                    .WORD   177777          ;DESTINATION WORD 2
6719 027360  000013                    .WORD   13              ;TEST PSW
6720 027362  177600                    .WORD   177600          ;RESULT WORD 1
6721 027364  000000                    .WORD   0               ;RESULT WORD 2
6722
6723 027366  177717                    .WORD   177717          ;SOURCE
6724 027370  177777                    .WORD   177777          ;DESTINATION WORD 1
6725 027372  000001                    .WORD   1               ;DESTINATION WORD 2
6726 027374  000011                    .WORD   11              ;TEST PSW
6727 027376  100000                    .WORD   100000          ;RESULT WORD 1
6728 027400  100000                    .WORD   100000          ;RESULT WORD 2
6729
6730 027402  177741                    .WORD   177741          ;SOURCE
6731 027404  100000                    .WORD   100000          ;DESTINATION WORD 1
6732 027406  000000                    .WORD   0               ;DESTINATION WORD 2
6733 027410  000010                    .WORD   10              ;TEST PSW
6734 027412  177777                    .WORD   177777          ;RESULT WORD 1
6735 027414  177777                    .WORD   177777          ;RESULT WORD 2
6736
6737 027416  177742                    .WORD   177742          ;SOURCE
6738 027420  037777                    .WORD   37777           ;DESTINATION WORD 1
6739 027422  177777                    .WORD   177777          ;DESTINATION WORD 2
6740 027424  000005                    .WORD   5               ;TEST PSW
6741 027426  000000                    .WORD   0               ;RESULT WORD 1
6742 027430  000000                    .WORD   0               ;RESULT WORD 2
6743
6744 027432  177742                    .WORD   177742          ;SOURCE
6745 027434  077777                    .WORD   77777           ;DESTINATION WORD 1
6746 027436  177777                    .WORD   177777          ;DESTINATION WORD 2
6747 027440  000001                    .WORD   1               ;TEST PSW
```

```
********** DOUBLE OPERAND TESTS ******************

 6748 027442  000000                    .WORD   0               ;RESULT WORD 1
 6749 027444  000001                    .WORD   1               ;RESULT WORD 2
 6750
 6751 027446  177711                    .WORD   177711          ;SOURCE
 6752 027450  065600                    .WORD   65600           ;DESTINATION WORD 1
 6753 027452  000012                    .WORD   12              ;DESTINATION WORD 2
 6754 027454  000003                    .WORD   3               ;TEST PSW
 6755 027456  000000                    .WORD   0               ;RESULT WORD 1
 6756 027460  012000                    .WORD   12000           ;RESULT WORD 2
 6757
 6758 027462  177740                    .WORD   177740          ;SOURCE
 6759 027464  077777                    .WORD   77777           ;DESTINATION WORD 1
 6760 027466  177777                    .WORD   177777          ;DESTINATION WORD 2
 6761 027470  000004                    .WORD   4               ;TEST PSW
 6762 027472  000000                    .WORD   0               ;RESULT WORD 1
 6763 027474  000000                    .WORD   0               ;RESULT WORD 2
 6764
 6765 027476  177737                    .WORD   177737          ;SOURCE
 6766 027500  177777                    .WORD   177777          ;DESTINATION WORD 1
 6767 027502  177774                    .WORD   177774          ;DESTINATION WORD 2
 6768 027504  000011                    .WORD   11              ;TEST PSW
 6769 027506  177777                    .WORD   177777          ;RESULT WORD 1
 6770 027510  177777                    .WORD   177777          ;RESULT WORD 2
 6771
 6772 027512  177747                    .WORD   177747          ;SOURCE
 6773 027514  100000                    .WORD   100000          ;DESTINATION WORD 1
 6774 027516  174000                    .WORD   174000          ;DESTINATION WORD 2
 6775 027520  000010                    .WORD   10              ;TEST PSW
 6776 027522  177777                    .WORD   177777          ;RESULT WORD 1
 6777 027524  177700                    .WORD   177700          ;RESULT WORD 2
 6778
 6779 027526  177753                    .WORD   177753          ;SOURCE
 6780 027530  006324                    .WORD   6324            ;DESTINATION WORD 1
 6781 027532  071002                    .WORD   71002           ;DESTINATION WORD 2
 6782 027534  000001                    .WORD   1               ;TEST PSW
 6783 027536  000000                    .WORD   0               ;RESULT WORD 1
 6784 027540  000146                    .WORD   146             ;RESULT WORD 2
 6785
 6786 027542  177765                    .WORD   177765          ;SOURCE
 6787 027544  102351                    .WORD   102351          ;DESTINATION WORD 1
 6788 027546  177231                    .WORD   177231          ;DESTINATION WORD 2
 6789 027550  000011                    .WORD   11              ;TEST PSW
 6790 027552  177760                    .WORD   177760          ;RESULT WORD 1
 6791 027554  116477                    .WORD   116477          ;RESULT WORD 2
 6792 027556                    FIN130:
 6795 027556                    MSPAU:
 6796
 6797                           ;       TEST THAT AUTO DEC/INC OPERATIONS USING SP ARE ON WORD BOUNDRYS
 6798 027556  005006                    CLR     R6                      ;CLEAR SP
 6799 027560  112667  153334            MOVB    (R6)+,COUNT             ;TRY AUTOINC ON R6
 6800 027564  022706  000002            CMP     #2,R6                   ;VERIFY AUTO INC BY 2
 6801 027570  001401                    BEQ     SPAU1                   ;BRANCH IF GOOD
 6802                                                                   ;BAD AUTO-INC
 6803 027572  104001                    ERROR   +1              ;CPU ERROR
 6804 027574  005006            SPAU1:  CLR     R6                      ;CLEAR R6
 6805 027576  112667  153316            MOVB    (R6)+,COUNT
 6806 027602  112667  153312            MOVB    (R6)+,COUNT             ;DOUBLE BYTE AUTO-INC
```

```
********** DOUBLE OPERAND TESTS ******************

 6807 027606  022706  000004                    CMP     #4,R6           ;VERIFY RESULT
 6808 027612  001401                            BEQ     SPAU2           ;BRANCH IF GOOD
 6809 027614  104001                            ERROR   +1              ;CPU ERROR
 6810                                                        ;BAD DOUBLE AUTO-INC
 6811 027616  012706  001000          SPAU2:    MOV     #STBOT,R6               ;LOAD R6
 6812 027622  114667  153272                    MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6813 027626  022706  000776                    CMP     #776,R6         ;VERIFY RESULT
 6814 027632  001401                            BEQ     SPAU3           ;BRANCH IF GOOD
 6815 027634  104001                            ERROR   +1              ;CPU ERROR
 6816
 6817 027636  012706  001000          SPAU3:    MOV     #STBOT,R6               ;LOAD R6
 6818 027642  114667  153252                    MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6819 027646  114667  153246                    MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6820 027652  022706  000774                    CMP     #774,R6         ;VERIFY RESULT
 6821 027656  001401                            BEQ     SPAU4           ;BRANCH IF GOOD
 6822 027660  104001                            ERROR   +1              ;CPU ERROR
 6823
 6824 027662  005006          SPAU4:    CLR     R6                      ; TEST AUTO-INC ON SOP
 6825 027664  105726                            TSTB    (R6)+           ; TEST AUTO-INC
 6826 027666  020627  000002                    CMP     R6,#2
 6827 027672  001401                            BEQ     SPAU5           ;BRANCH IF GOOD
 6828 027674  104001                            ERROR   +1              ;CPU ERROR
 6829
 6830 027676  012706  001000          SPAU5:    MOV     #STBOT,R6       ;LOAD R6
 6831 027702  105746                            TSTB    -(R6)           ; TEST AUTO-DEC
 6832 027704  022706  000776                    CMP     #776,R6         ;VERIFY RESULT
 6833 027710  001401                            BEQ     SPAU6           ;BRANCH IF GOOD
 6834 027712  104001                            ERROR   +1              ;CPU ERROR
 6835
 6836 027714  012706  001000          SPAU6:    MOV     #STBOT,R6
 6837
 6839                                ;
 6841 027720                         MTRY:
 6842
 6843                                ;       VERIFY YELLOW ZONE TRAP ON AUTO DEC OF R6
 6844 027720  005067  150042                    CLR     CPEREG          ;INIT CPU ERROR REGISTER
 6845 027724  012706  000150                    MOV     #150,R6         ;LOAD R6 WITH A VALUE THAT WILL
 6846                                                                   ;CAUSE A YELLOW STACK TRAP(IE. <400)
 6847 027730  016767  150050  153054           MOV     4,SLOC00        ;SAVE VECTOR
 6848 027736  012767  027774  150040           MOV     #MTRYA,4        ;SETUP THE STACK OVERFLOW TRAP POINTER
 6849 027744  016701  150176                    MOV     146,R1          ;SAVE VECTOR
 6850 027750  016702  150170                    MOV     144,R2          ;SAVE VECTOR
 6851 027754  016703  150162                    MOV     142,R3          ;SAVE VECTOR
 6852 027760  005067  150162                    CLR     146             ;JUST AS A PRECAUTION
 6853 027764  005046                            CLR     -(R6)           ;CAUSE A STACK OVERFLOW TRAP
 6854 027766  012706  001000                    MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
 6855 027772  104001                            ERROR   +1              ;CPU ERROR
 6856                                                        ;OVERFLOW TRAP FAILED
 6857 027774                         MTRYA:
 6858 027774  022767  000010  147764           CMP     #BIT03,CPEREG   ;WAS CPU ERROR REG SET PROPERLY?
 6859 030002  001003                            BNE     1$              ;GO TO ERROR IF NOT
 6860 030004  020627  000142                    CMP     R6,#142         ;VERIFY CORRECT DECRIMENT OF R6
 6861 030010  001401                            BEQ     MTRYB           ;BRANCH IF GOOD
 6862 030012  104001                  1$:       ERROR   +1              ;CPU ERROR
 6863                                                        ;R6 IMPROPERLY DECRIMENTED
 6864                                                        ;OR CPU ERROR REGISTER NOT CORRECT
 6865 030014                         MTRYB:
```

```
********** DOUBLE OPERAND TESTS ******************

 6866 030014  005067  147746                CLR     CPEREG          ;CLEAR THE CPU ERROR REGISTER
 6867 030020  016767  152766  147756        MOV     SLOC00,4        ;RESTORE VECTOR
 6868 030026  010167  150114                MOV     R1,146          ;RESTORE VECTORS
 6869 030032  010267  150106                MOV     R2,144          ;       "
 6870 030036  010367  150100                MOV     R3,142          ;       "
 6871 030042  012706  001000                MOV     #STBOT,R6
 6872
 6873
 6875                                ;
 6877 030046                         MTRYM:
 6878
 6879                                ;       TEST STACK OVERFLOW TRAPS IN VARIOUS MODES
 6880 030046  005067  147714                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6881 030052  012706  000400                MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6882 030056  016767  147722  152726        MOV     4,SLOC00                ;SAVE VECTOR
 6883 030064  012767  030106  147712        MOV     #TRYMA,4
 6884 030072  005067  150300                CLR     376                     ;JUST AS A PRECAUTION
 6885 030076  005046                        CLR     -(R6)                   ;CAUSE OVERFLOW TRAP
 6886 030100  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6887 030104  104001                        ERROR   +1              ;CPU ERROR
 6888                                                               ;NO OVERFLOW TRAP
 6889 030106                         TRYMA:
 6890 030106  005067  147654                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6891 030112  012705  001000                MOV     #1000,R5                ;SETUP R5 DATA
 6892 030116  012706  000400                MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6893 030122  012767  030140  147654        MOV     #TRYMB,4
 6894 030130  064645                        ADD     -(R6),-(R5)             ;CAUSE OVERFLOW TRAP
 6895 030132  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6896 030136  104001                        ERROR   +1              ;CPU ERROR
 6897                                                               ;NO OVERFLOW TRAP
 6898 030140                         TRYMB:
 6899 030140  005067  147622                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6900 030144  012706  000150                MOV     #150,R6                 ;SETUP OVERFLOW R6 DATA
 6901 030150  012767  030166  147626        MOV     #TRYMC,4
 6902 030156  044546                        BIC     -(R5),-(R6)             ;CAUSE OVERFLOW TRAP
 6903 030160  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6904 030164  104001                        ERROR   +1              ;CPU ERROR
 6905                                                               ;NO OVERFLOW TRAP
 6906 030166  005067  147574        TRYMC:  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6907 030172  016767  152614  147604        MOV     SLOC00,4                ;RESTORE VECTOR
 6908 030200  012706  001000                MOV     #STBOT,R6
 6909
 6910
 6912                                ;
 6914 030204                         MILLO:
 6915
 6916                                ;       TEST STACK OVERFLOW ON ILLEGAL INST TRAP
 6917 030204  005067  147556                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6918 030210  012706  000400                MOV     #400,R6                 ;SETUP FOR OVERFLOW TRAP
 6919 030214  016767  147570  152570        MOV     10,SLOC00               ;SAVE VECTOR
 6920 030222  012767  030250  147560        MOV     #MILLOA,10              ;SETUP ILLEGAL TRAP VECTOR
 6921 030230  016767  147550  152556        MOV     4,SLOC01                ;SAVE VECTOR
 6922 030236  012767  030256  147540        MOV     #MILLOB,4               ;SETUP OVERFLOW TRAP VECTOR
 6923 030244  000077                        77                              ;UNUSED INSTRUCTION TRAP
 6924 030246  000240                        NOP
 6925 030250  012706  001000        MILLOA: MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6926 030254  104001                        ERROR   +1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS ******************

 6927                                                          ;UNUSED INSTRUCTION TRAP
 6928 030256                                MILL08:
 6929 030256  016767  152532  147520              MOV     SLOC01,4                ;RESTORE VECTOR
 6930 030264  016767  152522  147516              MOV     SLOC00,10               ;RESTORE VECTOR
 6931 030272  005067  147470                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6932 030276  012706  001000                      MOV     #STBOT,R6               ;RESTORE R6
 6933
 6934
 6935
 6937                                       ;
 6939 030302                                MIOTO:
 6940
 6941                                       ;       TEST STACK OVERFLOW ON IOT TRAP
 6942 030302  005067  147460                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6943 030306  012706  000400                      MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
 6944 030312  016767  147502  152472              MOV     20,SLOC00                       ;SAVE OLD IOT VECTOR
 6945 030320  012767  030346  147472              MOV     #IOT0A,20                       ;SETUP ERROR ACTION ON IOT
 6946 030326  016767  147452  152460              MOV     4,SLOC01                        ;SAVE VECTOR
 6947 030334  012767  030354  147442              MOV     #IOT0B,4                        ;SETUP CORRECT TRAP VECTOR FOR
 6948                                                                                  ;OVERFLOW
 6949 030342  000004                              IOT                             ; TEST INSTRUCTION
 6950 030344  000240                              NOP
 6951 030346  012706  001000            IOT0A:    MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6952 030352  104001                              ERROR   +1              ;CPU ERROR
 6953                                                                             ;FAILURE OF STACK OVERFLOW
 6954 030354                                IOT0B:
 6955 030354  005067  147406                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6956 030360  012706  001000                      MOV     #STBOT,R6
 6957 030364  016767  152424  147412              MOV     SLOC01,4                        ;RESTORE VECTOR
 6958 030372  016767  152414  147420              MOV     SLOC00,20                       ;RESTORE TRAP VECTOR
 6959
 6961                                       ;
 6962                                       ;
 6964 030400                                MEMTO:
 6965
 6966                                       ;       TEST STACK OVERFLOW ON EMT TRAP
 6967 030400  005067  147362                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6968 030404  012706  000400                      MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
 6969 030410  016767  147414  152374              MOV     30,SLOC00                       ;SAVE OLD EMT VECTOR
 6970 030416  012767  030444  147404              MOV     #EMT0A,30                       ;SETUP ERROR ACTION ON EMT
 6971 030424  016767  147354  152362              MOV     4,SLOC01                        ;SAVE VECTOR
 6972 030432  012767  030452  147344              MOV     #EMT0B,4                        ;SETUP CORRECT TRAP VECTOR FOR
 6973                                                                                  ;OVERFLOW
 6974 030440  104000                              EMT                             ; TEST INSTRUCTION
 6975 030442  000240                              NOP
 6976 030444  012706  001000            EMT0A:    MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6977 030450  104001                              ERROR   +1              ;CPU ERROR
 6978                                                                             ;FAILURE OF STACK OVERFLOW
 6979 030452                                EMT0B:
 6980 030452  016767  152334  147350              MOV     SLOC00,30                       ;RESTORE TRAP VECTOR
 6981 030460  016767  152330  147316              MOV     SLOC01,4                        ;RESTORE VECTOR
 6982 030466  005067  147274                      CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6983 030472  012706  001000                      MOV     #STBOT,R6
 6984
 6987 030476                                MTRP0:
 6988
 6989                                       ;       TEST STACK OVERFLOW ON TRAP
```

```
********** DOUBLE OPERAND TESTS ******************

6990 030476  005067  147264                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
6991 030502  012706  000400                    MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
6992 030506  016767  147322  152276            MOV     34,SLOC00                       ;SAVE OLD TRP VECTOR
6993 030514  012767  030542  147312            MOV     #TRP0A,34                       ;SETUP ERROR ACTION ON TRP
6994 030522  016767  147256  152264            MOV     4,SLOC01                        ;SAVE VECTOR
6995 030530  012767  030550  147246            MOV     #TRP0B,4                        ;SETUP CORRECT TRAP VECTOR FOR
6996                                                                                    ;OVERFLOW
6997 030536  104400                            TRAP                                    ; TEST INSTRUCTION
6998 030540  000240                            NOP
6999 030542  012706  001000          TRP0A:    MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
7000 030546  104001                            ERROR   +1              ;CPU ERROR
7001                                                                    ;FAILURE OF STACK OVERFLOW
7002 030550                          TRP0B:
7003 030550  016767  152236  147256            MOV     SLOC00,34                       ;RESTORE TRAP VECTOR
7004 030556  016767  152232  147220            MOV     SLOC01,4                        ;RESTORE VECTOR
7005 030564  005067  147176                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7006 030570  012706  001000                    MOV     #STBOT,R6
7007
7009                                 ;
7010                                 ;
7012 030574                          MBPT0:
7013
7014                                 ;       TEST STACK OVERFLOW ON BPT
7015 030574  005067  147166                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7016 030600  012706  000400                    MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
7017 030604  016767  147204  152200            MOV     14,SLOC00                       ;SAVE OLD BPT VECTOR
7018 030612  012767  030640  147174            MOV     #BPT0A,14                       ;SETUP ERROR ACTION ON BPT
7019 030620  016767  147160  152166            MOV     4,SLOC01                        ;SAVE VECTOR
7020 030626  012767  030646  147150            MOV     #BPT0B,4                        ;SETUP CORRECT TRAP VECTOR FOR
7021                                                                                    ;OVERFLOW
7022 030634  000003                            BPT                                     ; TEST INSTRUCTION
7023 030636  000240                            NOP
7024 030640  012706  001000          BPT0A:    MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
7025 030644  104001                            ERROR   +1              ;CPU ERROR
7026                                                                    ;FAILURE OF STACK OVERFLOW
7027 030646                          BPT0B:
7028 030646  005067  147114                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7029 030652  016767  152134  147134            MOV     SLOC00,14                       ;RESTORE TRAP VECTOR
7030 030660  016767  152130  147116            MOV     SLOC01,4                        ;RESTORE VECTOR
7031 030666  012706  001000                    MOV     #STBOT,R6
7032
7034                                 ;
7035                                 ;
7037 030672                          MILA0:
7038
7039                                 ;       TEST STACK OVERFLOW AND ILLEGAL JMP INSTRUCTION
7040 030672  005067  147070                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7041 030676  012706  000400                    MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
7042 030702  016767  147102  152102            MOV     10,SLOC00                       ;SAVE OLD ILLEGAL INST. VECTOR
7043 030710  012767  030740  147072            MOV     #ILA0A,10                       ;SETUP ERROR ACTION ILLEGAL OPCODE
7044 030716  016767  147062  152070            MOV     4,SLOC01                        ;SAVE VECTOR
7045 030724  012767  030746  147052            MOV     #ILB0B,4                        ;SETUP CORRECT TRAP VECTOR FOR
7046                                                                                    ;OVERFLOW
7047 030732  005001                            CLR     R1
7048 030734  000101                            JMP     R1                              ; TEST INSTRUCTION
7049 030736  000240                            NOP
7050 030740  012706  001000          ILA0A:    MOV     #STBOT,R6                       ;RESTORE R6 FOR ERROR CALL
```

```
********** DOUBLE OPERAND TESTS ******************

 7051 030744  104001                          ERROR   +1              ;CPU ERROR
 7052                                                                 ;FAILURE OF STACK OVERFLOW
 7053 030746                            ILLBOB:
 7054 030746  016767  152042  147030          MOV     SLOC01,4                ;RESTORE VECTOR
 7055 030754  016767  152032  147026          MOV     SLOC00,10               ;RESTORE TRAP VECTOR
 7056 030762  005067  147000                  CLR     CPEREG          ;CLEAR CPU ERROR REGISTER
 7057 030766  012706  001000                  MOV     #STBOT,R6
 7058
 7060                                   ;
 7061                                   ;
 7063 030772                            MILLBO:
 7064
 7065                                   ;       TEST STACK OVERFLOW ON ILLEGAL JSR INST
 7066 030772  012706  000400                  MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
 7067 030776  016767  147006  152006          MOV     10,SLOC00               ;SAVE OLD VECTOR
 7068 031004  012767  031034  146776          MOV     #ILLBOA,10              ;SETUP ERROR ACTION ON ILL. OPCODE
 7069 031012  016767  146766  151774          MOV     4,SLOC01                ;SAVE VECTOR
 7070 031020  012767  031042  146756          MOV     #ILLBOB,4               ;SETUP CORRECT TRAP VECTOR FOR
 7071                                                                         ;OVERFLOW
 7072 031026  005001                          CLR     R1
 7073 031030  004501                          JSR     R5,R1                   ; TEST INSTRUCTION
 7074 031032  000240                          NOP
 7075 031034  012706  001000            ILLBOA: MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 7076 031040  104001                          ERROR   +1              ;CPU ERROR
 7077                                                                 ;FAILURE OF STACK OVERFLOW
 7078 031042                            ILLBOB:
 7079 031042  016767  151746  146734          MOV     SLOC01,4                ;RESTORE VECTOR
 7080 031050  016767  151736  146732          MOV     SLOC00,10               ;RESTORE TRAP VECTOR
 7081 031056  012706  001000                  MOV     #STBOT,R6
 7082
 7084                                   ;
 7085                                   ;
 7087 031062                            MSTO:
 7088
 7089                                   ;       TEST FOR FALSE STACK OVERFLOW
 7090 031062  016767  146716  151722          MOV     4,SLOC00                ;SAVE VECTOR
 7091 031070  012767  031136  146706          MOV     #MSTOE,4                ;ANTICIPATE OVERFLOW ERROR
 7092 031076  012706  001002                  MOV     #1002,R6                ;SETUP LEGAL R6
 7093 031102  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
 7094 031104  012706  002002                  MOV     #2002,R6                ;SETUP LEGAL R6
 7095 031110  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
 7096 031112  012706  004002                  MOV     #4002,R6                ;SETUP LEGAL R6
 7097 031116  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
 7098 031120  012706  010002                  MOV     #10002,R6               ;SETUP LEGAL R6
 7099 031124  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
 7100 031126  012706  100402                  MOV     #100402,R6              ;SETUP  LEGAL R6
 7101 031132  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
 7102 031134  000403                          BR      MSTOEE                  ;EXIT MODULE
 7103 031136  012706  001000            MSTOE:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 7104 031142  104001                          ERROR   +1              ;CPU ERROR
 7105                                                                 ;STACK OVERFLOW ERROR
 7106 031144  016767  151642  146632  MSTOEE: MOV     SLOC00,4                ;RESTORE VECTOR
 7107 031152  012706  001000                  MOV     #STBOT,R6
 7108
 7110                                   ;
 7111                                   ;
 7113 031156                            MTT:
```

********** DOUBLE OPERAND TESTS **********

```
7114
7115                                    ;       TEST T-BIT TRAPS
7116 031156  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7117 031162  016767  146626  151622            MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7118 031170  012746  000020                    MOV     #20,-(R6)               ;PUSH T-BIT
7119 031174  012746  031212                    MOV     #MTTA,-(R6)             ;SETUP ERROR TRAP VECTOR
7120 031200  012767  031214  146606            MOV     #MTTB,14                ;SETUP NEW T-BIT VECTOR
7121 031206  000002                            RTI                             ;CAUSE A T BIT SET IN PSW
7122 031210  104001                            ERROR   .1              ;CPU ERROR
7123                                                                    ;SHOULD NEVER BE EXECUTED
7124 031212  104001                    MTTA:   ERROR   .1              ;CPU ERROR
7125                                                                    ;DIDNT TAKE CORRECT TRAP
7126 031214  022706  000774            MTTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
7127 031220  001401                            BEQ     MTTD                    ;BRANCH IF GOOD
7128 031222  104001                            ERROR   .1              ;CPU ERROR
7129                                                                    ;BAD SP
7130 031224  021627  031212            MTTD:   CMP     (R6),#MTTA              ;VERIFY PC SAVED ON STACK
7131 031230  001401                            BEQ     MTTE                    ;BRANCH IF GOOD
7132 031232  104001                            ERROR   .1              ;CPU ERROR
7133                                                                    ;INCORRECT PC ON STACK
7134 031234                            MTTE:
7135 031234  016767  151552  146552            MOV     SLOC00,14               ;RESTORE VECTOR 14
7136
7137 031242  012706  001000                    MOV     #STBOT,R6
7138
7140                                    ;
7142 031246                            MTTS:
7143
7144                                    ;       TEST T-BIT TRAPS WITH RTT
7145 031246  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7146 031252  016767  146536  151532            MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7147 031260  012746  000020                    MOV     #20,-(R6)               ;PUSH T-BIT
7148 031264  012746  031302                    MOV     #MTTSA,-(R6)            ;SETUP ERROR TRAP VECTOR
7149 031270  012767  031306  146516            MOV     #MTTSB,14               ;SETUP NEW T-BIT VECTOR
7150 031276  000006                            RTT                             ;CAUSE A T BIT SET IN PSW
7151 031300  104001                            ERROR   .1              ;CPU ERROR
7152                                                                    ;SHOULD NEVER BE EXECUTED
7153 031302  000240                    MTTSA:  NOP                             ;RTT WILL EXECUTE THIS INSTRUCTION
7154                                                                    ;WITH A T-BIT TRAP
7155 031304  104001                    MTTSQ:  ERROR   .1              ;CPU ERROR
7156                                                                    ;DIDNT TAKE CORRECT TRAP
7157 031306  022706  000774            MTTSB:  CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
7158 031312  001401                            BEQ     MTTSD                   ;BRANCH IF GOOD
7159 031314  104001                            ERROR   .1              ;CPU ERROR
7160                                                                    ;BAD SP
7161 031316  021627  031304            MTTSD:  CMP     (R6),#MTTSQ             ;VERIFY PC SAVED ON STACK
7162 031322  001401                            BEQ     MTTSE                   ;BRANCH IF GOOD
7163 031324  104001                            ERROR   .1              ;CPU ERROR
7164                                                                    ;INCORRECT PC ON STACK
7165 031326                            MTTSE:
7166 031326  016767  151460  146460            MOV     SLOC00,14               ;RESTORE VECTOR 14
7167 031334  012706  001000                    MOV     #STBOT,R6
7168
7171 031340                            MTTR:
7172
7173                                    ;       TEST OLD STATUS ON T-BIT TRAP
7174 031340  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
```

```
********** DOUBLE OPERAND TESTS **********

7175 031344  016767  146444  151440          MOV     14,SLOC00               ;SAVE OLD  VECTOR
7176 031352  012746  000020                  MOV     #20,-(R6)               ;PUSH T-BIT
7177 031356  012746  031404                  MOV     #MTTRA,-(R6)            ;SETUP ERROR TRAP VECTOR
7178 031362  012767  031406  146424          MOV     #MTTRB,14               ;SETUP NEW T-BIT VECTOR
7179 031370  012767  000357  146400          MOV     #357,PS                 ;SET PRIORITY AND COND C
7180 031376  000277                          SCC
7181 031400  000002                          RTI
7182 031402  104001                          ERROR   .1              ;CPU ERROR
7183                                                                 ;SHOULD NEVER EXECUTE
7184 031404  104001                  MTTRA:  ERROR   .1              ;CPU ERROR
7185                                                                 ;DIDNT TAKE CORRECT TRAP
7186 031406  026727  147364  000020  MTTRB:  CMP     STBOT-2,#20             ;VERIFY PSW ON STACK
7187 031414  001401                          BEQ     MTTRC                   ;BRANCH IF CORRECT STATUS
7188 031416  104001                          ERROR   .1              ;CPU ERROR
7189                                                                 ;BAD STATUS ON STACK
7190 031420  012706  001000          MTTRC:  MOV     #STBOT,R6               ;SETUP STACK
7191 031424  012746  000377                  MOV     #377,-(R6)              ;PUSH T-BIT
7192 031430  012746  031456                  MOV     #MTTRD,-(R6)            ;SETUP ERROR TRAP VECTOR
7193 031434  012767  031460  146352          MOV     #MTTRE,14               ;SETUP NEW T-BIT VECTOR
7194 031442  012767  000000  146326          MOV     #0,PS                   ;CLEAR PRIORITY
7195 031450  000257                          CCC                             ;CLEAR CONDITION CODES
7196 031452  000002                          RTI
7197 031454  104001                          ERROR   .1              ;CPU ERROR
7198                                                                 ;SHOULD NEVER EXECUTE
7199 031456  104001                  MTTRD:  ERROR   .1              ;CPU ERROR
7200                                                                 ;DIDNT TAKE CORRECT TRAP
7201 031460  026727  147312  000377  MTTRE:  CMP     STBOT-2,#377            ;VERIFY OLD PSW ON STACK
7202 031466  001401                          BEQ     MTTRF                   ;BRANCH IF GOOD
7203 031470  104001                          ERROR   .1              ;CPU ERROR
7204                                                                 ;OLD PSW INCORRECT
7205 031472                          MTTRF:
7206 031472  016767  151314  146314          MOV     SLOC00,14               ;RESTORE VECTOR
7207 031500  012706  001000                  MOV     #STBOT,R6
7208
7210                                 ;
7212 031504                          MRT:
7213
7214                                 ;       TEST RESERVED INST TRAP
7215 031504  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7216 031510  016767  146274  151274          MOV     10,SLOC00               ;SAVE OLD  VECTOR
7217 031516  012767  031530  146264          MOV     #MRTB,10                ;SETUP NEW RESERVED VECTOR
7218 031524  000077                          77
7219 031526  104001                  MRTA:   ERROR   .1              ;CPU ERROR
7220                                                                 ;DIDNT TAKE CORRECT TRAP
7221 031530  022706  000774          MRTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7222 031534  001401                          BEQ     MRTE                    ;BRANCH IF GOOD
7223 031536  104001                          ERROR   .1              ;CPU ERROR
7224                                                                 ;BAD PC ON STACK
7225 031540  021627  031526          MRTE:   CMP     (R6),#MRTA              ;VERIFY PROPER PC ON STACK
7226 031544  001401                          BEQ     MRTF                    ;BRANCH IF GOOD
7227 031546  104001                          ERROR   .1              ;CPU ERROR
7228                                                                 ;INCORRECT PC ON STACK
7229 031550                          MRTF:
7230 031550  016767  151236  146232          MOV     SLOC00,10               ;RESTORE TRAP VECTOR
7231 031556  012706  001000                  MOV     #STBOT,R6
7232
7234                                 ;
```

********** DOUBLE OPERAND TESTS **********

```
7235                                    ;
7237 031562                             MRTO:
7238
7239                                    ;       TEST OLD STATUS ON RESERVED INST TRAP
7240 031562  012706  001000                        MOV     #STBOT,R6               ;SETUP STACK
7241 031566  016767  146216  151216               MOV     10,SLOC00               ;SAVE OLD  VECTOR
7242 031574  012767  031614  146206               MOV     #MRTOB,10               ;SETUP NEW VECTOR
7243 031602  005067  146170                        CLR     PS                      ;CLEAR PRIORITY AND COND C
7244 031606  000257                               CCC
7245 031610  000077                               77
7246 031612  104001                     MRTOA:    ERROR   .1              ;CPU ERROR
7247                                                                       ;DIDNT TAKE CORRECT TRAP
7248 031614  026727  147156  000000     MRTOB:    CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7249 031622  001401                               BEQ     MRTOC                   ;BRANCH IF CORRECT STATUS
7250 031624  104001                               ERROR   .1              ;CPU ERROR
7251                                                                       ;BAD STATUS ON STACK
7252 031626  012706  001000     MRTOC:    MOV     #STBOT,R6               ;SETUP STACK
7253 031632  012767  031654  146150               MOV     #MRTOE,10               ;SET UP TRAP VECTOR
7254 031640  012767  000357  146130               MOV     #357,PS                 ;SET PRIORITY
7255 031646  000277                               SCC                             ;SET CONDITION CODES
7256 031650  000077                               77                              ;RESERVED INSTRUCTION
7257
7258 031652  104001                     MRTOD:    ERROR   .1              ;CPU ERROR
7259                                                                       ;DIDNT TAKE CORRECT TRAP
7260 031654  026727  147116  000357     MRTOE:    CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7261 031662  001401                               BEQ     MRTOF                   ;BRANCH IF GOOD
7262 031664  104001                               ERROR   .1              ;CPU ERROR
7263                                                                       ;OLD PSW INCORRECT
7264 031666                             MRTOF:
7265 031666  016767  151120  146114               MOV     SLOC00,10               ;RESOTRE TRAP VECTOR
7266 031674  012706  001000                        MOV     #STBOT,R6
7267
7270 031700                             MTP:
7271
7272                                    ;       TEST TRAP INST
7273 031700  012706  001000                        MOV     #STBOT,R6               ;SETUP STACK
7274 031704  016767  146124  151100               MOV     34,SLOC00               ;SAVE OLD  VECTOR
7275 031712  012767  031732  146114               MOV     #MTPB,34                ;SETUP NEW TRAP VECTOR
7276 031720  005067  146052                        CLR     PS                      ;CLEAR PRIORITY ABND COND C
7277 031724  000257                               CCC
7278 031726  104400                               TRAP
7279 031730  104001                     MTPR:     ERROR   .1              ;CPU ERROR
7280                                                                       ;DIDNT TAKE CORRECT TRAP
7281 031732  022706  000774     MTPB:     CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7282 031736  001401                               BEQ     MTPQ                    ;BRANCH IF GOOD
7283 031740  104001                               ERROR   .1              ;CPU ERROR
7284                                                                       ;BAD PC ON STACK
7285 031742  021627  031730     MTPQ:     CMP     (R6),#MTPR              ;VERFY PROPER PC ON STACK
7286 031746  001401                               BEQ     MTPF                    ;BRANCH IF GOOD
7287 031750  104001                               ERROR   .1              ;CPU ERROR
7288                                                                       ;INCORRECT PC ON STACK
7289 031752                             MTPF:
7290 031752  016767  151034  146054               MOV     SLOC00,34               ;RESTORE VECTOR
7291 031760  012706  001000                        MOV     #STBOT,R6
7292
7294                                    ;
7295                                    ;
```

```
********** DOUBLE OPERAND TESTS ******************

7297 031764                          MTPO:
7298
7299                                 ;       TEST OLD STATUS SAVED ON TRAP
7300 031764  012706  001000                     MOV     #STBOT,R6               ;SETUP STACK
7301 031770  016767  146040  151014             MOV     34,SLOC00               ;SAVE OLD  VECTOR
7302 031776  012767  032016  146030             MOV     #MTPOB,34               ;SETUP NEW TRAP VECTOR
7303 032004  005067  145766                     CLR     PS                      ;CLEAR PRIORITY AND COND C
7304 032010  000257                             CCC
7305 032012  104400                             TRAP
7306 032014  104001                  MTPOA:     ERROR   +1              ;CPU ERROR
7307                                                                     ;DIDNT TAKE CORRECT TRAP
7308 032016  026727  146754  000000  MTPOB:     CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7309 032024  001401                             BEQ     MTPOC                   ;BRANCH IF CORRECT STATUS
7310 032026  104001                             ERROR   +1              ;CPU ERROR
7311                                                                     ;BAD STATUS ON STACK
7312 032030  012706  001000          MTPOC:     MOV     #STBOT,R6               ;SETUP STACK
7313 032034  012767  032056  145772             MOV     #MTPOE,34               ;SET UP TRAP VECTOR
7314 032042  012767  000357  145726             MOV     #357,PS                 ;SET PRIORITY
7315 032050  000277                             SCC                             ;SET CONDITION CODES
7316 032052  104400                             TRAP                            ;ISSUE TRAP
7317 032054  104001                  MTPOD:     ERROR   +1              ;CPU ERROR
7318                                                                     ;DIDNT TAKE CORRECT TRAP
7319 032056  026727  146714  000357  MTPOE:     CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7320 032064  001401                             BEQ     MTPOF                   ;BRANCH IF GOOD
7321 032066  104001                             ERROR   +1              ;CPU ERROR
7322                                                                     ;OLD PSW INCORRECT
7323 032070                          MTPOF:
7324 032070  016767  150716  145736             MOV     SLOC00,34               ;RESTORE TRAP VECTOR
7325 032076  012706  001000                     MOV     #STBOT,R6
7326
7328                                 ;
7329                                 ;
7331 032102                          MTPA:
7332
7333                                 ;       TEST ALL TRAP OPCODES - SELF MODIFYING
7334 032102  005003                             CLR     R3                      ;SETUP REGISTER TO INDICATE OPCODE
7335 032104  012706  001000                     MOV     #STBOT,R6               ;SETUP STACK
7336 032110  016767  145720  150674             MOV     34,SLOC00               ;SAVE OLD  VECTOR
7337 032116  016767  145662  150670             MOV     4,SLOC01                ;SAVE IN CASE OF HALT
7338 032124  012767  032154  145652             MOV     #MTPAH,4                ;SETUP HALT TRAP
7339 032132  012767  032156  145674             MOV     #MTPAA,34               ;SETUP NEW TRAP VECTOR
7340 032140  000167  000012                     JMP     MTPAA                   ;GO INTO LOOPING CODE
7341                                 ;
7342 032144  000000                  MTPAL:     HALT                            ;SET TO A ZERO
7343 032146  104001                             ERROR   +1              ;CPU ERROR
7344                                                                     ;TRAP INSTRUCTION FAILED TO TRAP
7345                                                                             ;EXAMINE OPCCODE AT LOCATION MTPAL:
7346 032150  000167  000002                     JMP     MTPAA                   ;ATTEMPT TO GO ON
7347                                 ;
7348 032154  104001                  MTPAH:     ERROR   +1              ;CPU ERROR
7349                                                                     ;ERROR, EITHER CANT MODIFY LOCATION MTPAL
7350                                                                             ;OR TRAP INSTRUCTION FAILED
7351 032156                          MTPAA:
7352 032156  005203                             INC     R3                      ;GET NEXT OPCODE
7353
7354 032160  012706  001000                     MOV     #STBOT,R6               ;RESTORE STACK
7355 032164  020327  000400                     CMP     R3,#400                 ;SEE IF LAST OPCODE
```

```
032170  001406                          BEQ     MTPAE                   ;BRANCH IF DONE
032172  012767  104400  177744          MOV     #104400,MTPAL           ;TRAP OPCODE INTO LOCATION
032200  060367  177740                  ADD     R3,MTPAL                ;FORM TEST OPCODE
032204  000757                          BR      MTPAL                   ;EXECUTE TEST
032206                          MTPAE:

032206  016767  150600  145620          MOV     SLOC00,34
032214  016767  150574  145562          MOV     SLOC01,4                ;RESTORE VECTORS

032222  012706  001000                  MOV     #STBOT,R6

                                ;
                                ;
032226                          MIOT:

                                ;       TEST IOT TRAP
032226  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
032232  016767  145562  150552          MOV     20,SLOC00               ;SAVE OLD  VECTOR
032240  012767  032252  145552          MOV     #MIOTB,20               ;SETUP NEW IOT VECTOR
032246  000004                          IOT
032250  104001                  MIOTA:  ERROR   +1                ;CPU ERROR
                                                                  ;DIDNT TAKE CORRECT TRAP
032252  022706  000774          MIOTB:  CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
032256  001401                          BEQ     MIOTD                   ;BRANCH IF GOOD
032260  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;BAD PC ON STACK
032262  021627  032250          MIOTD:  CMP     (R6),#MIOTA             ;VERFY PROPER PC ON STACK
032266  001401                          BEQ     MIOTF                   ;BRANCH IF GOOD
032270  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;INCORRECT PC ON STACK
032272  016767  150514  145520  MIOTF:  MOV     SLOC00,20               ;RESTORE VECTOR
032300  012706  001000                  MOV     #STBOT,R6

032304                          MITO:

                                ;       TEST OLD STATUS ON IOT TRAP
032304  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
032310  016767  145504  150474          MOV     20,SLOC00               ;SAVE OLD  VECTOR
032316  012767  032336  145474          MOV     #MITOB,20               ;SETUP NEW IOT VECTOR
032324  005067  145446                  CLR     PS                      ;CLEAR PRIORITY AND COND C
032330  000257                          CCC
032332  000004                          IOT
032334  104001                  MITOA:  ERROR   +1                ;CPU ERROR
                                                                  ;DIDNT TAKE CORRECT TRAP
032336  026727  146434  000000  MITOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
032344  001401                          BEQ     MITOC                   ;BRANCH IF CORRECT STATUS
032346  104001                          ERROR   +1                ;CPU ERROR
                                                                  ;BAD STATUS ON STACK
032350  012706  001000          MITOC:  MOV     #STBOT,R6               ;SETUP STACK
032354  012767  032376  145436          MOV     #MITOE,20               ;SET UP TRAP VECTOR
032362  012767  000357  145406          MOV     #357,PS                 ;SET PRIORITY
032370  000277                          SCC                             ;SET CONDITION CODES
032372  000004                          IOT
032374  104001                  MITOD:  ERROR   +1                ;CPU ERROR
                                                                  ;DIDNT TAKE CORRECT TRAP
032376  026727  146374  000357  MITOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
032404  001401                          BEQ     MITOF                   ;BRANCH IF GOOD
```

```
********** DOUBLE OPERAND TESTS ******************

7417 032406  104001                          ERROR   +1              ;CPU ERROR
7418                                                                 ;OLD PSW INCORRECT
7419 032410                            MITOF:
7420 032410  016767  150376  145402          MOV     SLOC00,20               ;RESTORE VECTOR
7421 032416  012706  001000                   MOV     #STBOT,R6
7422
7424                                   ;
7426 032422                            MET:
7427
7428                                   ;       TEST EMULATOR TRAP INSTRUCTION  (EMT)
7429 032422  012706  001000                   MOV     #STBOT,R6               ;SETUP STACK
7430 032426  016767  145376  150356          MOV     30,SLOC00               ;SAVE OLD  VECTOR
7431 032434  012767  032470  145366          MOV     #METB,30                ;SETUP NEW EMT VECTOR
7432 032442  016767  145366  150344          MOV     34,SLOC01               ;SAVE TRAP VECTOR
7433 032450  012767  136104  145356          MOV     #$ERROR,34              ;SET UP TO HANDLE EMT ERROR
7434 032456  104000                          EMT
7435 032460  104400                    META:  TRAP                            ;TRAP ON ERROR
7436 032462  001057                          .WORD   559.
7437 032464  000001                          .WORD   1                       ;CPUERR
7438 032466  000001                          .WORD   1                       ;ERRTN
7439                                                                 ;DIDNT TAKE CORRECT TRAP
7440 032470  022706  000774            METB:  CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7441 032474  001401                          BEQ     METD                    ;BRANCH IF GOOD
7442 032476  104001                          ERROR   +1              ;CPU ERROR
7443                                                                 ;BAD PC ON STACK
7444 032500  021627  032460            METD:  CMP     (R6),#META              ;VERFY PROPER PC ON STACK
7445 032504  001401                          BEQ     METF                    ;BRANCH IF GOOD
7446 032506  104001                          ERROR   +1              ;CPU ERROR
7447                                                                 ;INCORRECT PC ON STACK
7448 032510  016767  150300  145316  METF:   MOV     SLOC01,34               ;RESTORE VECTOR
7449 032516  016767  150270  145304          MOV     SLOC00,30               ;RESTORE VECTOR
7450 032524  012706  001000                   MOV     #STBOT,R6
7451
7453                                   ;
7455 032530                            METO:
7456
7457                                   ;       TEST OLD STATUS ON EMT TRAP
7458 032530  012706  001000                   MOV     #STBOT,R6               ;SETUP STACK
7459 032534  016767  145270  150250          MOV     30,SLOC00               ;SAVE OLD  VECTOR
7460 032542  012767  032604  145260          MOV     #METOB,30               ;SETUP NEW EMT VECTOR
7461 032550  016767  145260  150236          MOV     34,SLOC01               ;SAVE TRAP VECTOR
7462 032556  012767  136104  145250          MOV     #$ERROR,34              ;SET UP TRAP VECTOR
7463 032564  005067  145206                   CLR     PS                      ;CLEAR PRIORITY AND COND C
7464 032570  000257                          CCC
7465 032572  104000                          EMT
7466 032574  104400                    METOA: TRAP
7467 032576  001062                          .WORD   562.
7468 032600  000001                          .WORD   1                       ;CPUERR
7469 032602  000001                          .WORD   1                       ;ERRTN
7470                                                                 ;DIDNT TAKE CORRECT TRAP
7471 032604  026727  146166  000000  METOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7472 032612  001401                          BEQ     METOC                   ;BRANCH IF CORRECT STATUS
7473 032614  104001                          ERROR   +1              ;CPU ERROR
7474                                                                 ;BAD STATUS ON STACK
7475 032616  012706  001000            METOC: MOV     #STBOT,R6               ;SETUP STACK
7476 032622  012767  032652  145200          MOV     #METOE,30               ;SET UP TRAP VECTOR
7477 032630  012767  000357  145140          MOV     #357,PS                 ;SET PRIORITY
```

```
032636  000277                          SCC                                     ;SET CONDITION CODES
032640  104000                          EMT
032642  104400                  METOD:  TRAP
032644  001064                          .WORD   564.
032646  000001                          .WORD   1                               ;CPUERR
032650  000001                          .WORD   1                               ;ERRTN
                                                                        ;DIDNT TAKE CORRECT TRAP
032652  026727  146120  000357  METOE:  CMP     STBOT-2,#357                    ;VERIFY OLD PSW ON STACK
032660  001401                          BEQ     METOF                           ;BRANCH IF GOOD
032662  104001                          ERROR   +1                      ;CPU ERROR
                                                                        ;OLD PSW INCORRECT
032664                          METOF:
032664  016767  150124  145142          MOV     SLOC01,34                       ;RESTORE VECTOR
032672  016767  150114  145130          MOV     SLOC00,30                       ;RESTORE VECTOR
032700  012706  001000                  MOV     #STBOT,R6

                                ;
                                ;
032704                          MBT:

                                ;       TEST BPT TRAP
032704  012706  001000                  MOV     #STBOT,R6                       ;SETUP STACK
032710  016767  145100  150074          MOV     14,SLOC00                       ;SAVE OLD  VECTOR
032716  012767  032730  145070          MOV     #MBTB,14                        ;SETUP NEW BPT VECTOR
032724  000003                          BPT
032726  104001                  MBTA:   ERROR   +1                      ;CPU ERROR
                                                                        ;DIDNT TAKE CORRECT TRAP
032730  022706  000774          MBTB:   CMP     #STBOT-4,R6                     ;VERIFY SP DECRIMENT
032734  001401                          BEQ     MBTD                            ;BRANCH IF GOOD
032736  104001                          ERROR   +1                      ;CPU ERROR
                                                                        ;BAD PC ON STACK
032740  021627  032726          MBTD:   CMP     (R6),#MBTA                      ;VERFY PROPER PC ON STACK
032744  001401                          BEQ     MBTF                            ;BRANCH IF GOOD
032746  104001                          ERROR   +1                      ;CPU ERROR
                                                                        ;INCORRECT PC ON STACK
032750  016767  150036  145036  MBTF:   MOV     SLOC00,14                       ;RESTORE VECTOR
032756  012706  001000                  MOV     #STBOT,R6

                                ;
                                ;
032762                          MBTO:

                                ;       TEST OLD STATUS ON BPT TRAP
032762  012706  001000                  MOV     #STBOT,R6                       ;SETUP STACK
032766  016767  145022  150016          MOV     14,SLOC00                       ;SAVE OLD  VECTOR
032774  012767  033014  145012          MOV     #MBTOB,14                       ;SETUP NEW BPT VECTOR
033002  005067  144770                  CLR     PS                              ;CLEAR PRIORITY AND COND C
033006  000257                          CCC
033010  000003                          BPT
033012  104001                  MBTOA:  ERROR   +1                      ;CPU ERROR
                                                                        ;DIDNT TAKE CORRECT TRAP
033014  026727  145756  000000  MBTOB:  CMP     STBOT-2,#0                      ;VERIFY PSW ON STACK
033022  001401                          BEQ     MBTOC                   ;BRANCH IF CORRECT STATUS
033024  104001                          ERROR   +1                      ;CPU ERROR
                                                                        ;BAD STATUS ON STACK
033026  012706  001000          MBTOC:  MOV     #STBOT,R6                       ;SETUP STACK
033032  012767  033054  144754          MOV     #MBTOE,14                       ;SET UP TRAP VECTOR
```

```
********** DOUBLE OPERAND TESTS ******************

 7539 033040  012767  000357  144730          MOV     #357,PS                 ;SET PRIORITY
 7540 033046  000277                          SCC                             ;SET CONDITION CODES
 7541 033050  000003                          BPT
 7542 033052  104001                  MBTOD:  ERROR   +1              ;CPU ERROR
 7543                                                                 ;DIDNT TAKE CORRECT TRAP
 7544 033054  026727  145716  000357  MBTOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
 7545 033062  001401                          BEQ     MBTOF                   ;BRANCH IF GOOD
 7546 033064  104001                          ERROR   +1              ;CPU ERROR
 7547                                                                 ;OLD PSW INCORRECT
 7548 033066                          MBTOF:
 7549 033066  016767  147720  144720          MOV     SLOC00,14               ;RESTORE VECTOR
 7550 033074  012706  001000                  MOV     #STBOT,R6
 7551
 7553                                 ;
 7554                                 ;
 7556 033100                          MIL:
 7557
 7558                                 ;       TEST ILLEGAL JUMP INSTRUCTION TRAP
 7559 033100  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7560 033104  016767  144700  147700          MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7561 033112  012767  033126  144670          MOV     #MILB,10                ;SETUP NEW ILLEGAL VECTOR
 7562 033120  005001                          CLR     R1
 7563 033122  000101                          JMP     R1                      ;**TEST INSTRUCTIO
 7564 033124  104001                  MILA:   ERROR   +1              ;CPU ERROR
 7565                                                                 ;DIDNT TAKE CORRECT TRAP
 7566 033126  022706  000774          MILB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
 7567 033132  001401                          BEQ     MILD                    ;BRANCH IF GOOD
 7568 033134  104001                          ERROR   +1              ;CPU ERROR
 7569                                                                 ;BAD PC ON STACK
 7570 033136  021627  033124          MILD:   CMP     (R6),#MILA              ;VERFY PROPER PC ON STACK
 7571 033142  001401                          BEQ     MILF                    ;BRANCH IF GOOD
 7572 033144  104001                          ERROR   +1              ;CPU ERROR
 7573                                                                 ;INCORRECT PC ON STACK
 7574 033146  016767  147640  144634  MILF:   MOV     SLOC00,10               ;RESTORE VECTOR
 7575 033154  012706  001000                  MOV     #STBOT,R6
 7576
 7579 033160                          MILO:
 7580
 7581                                 ;       TEST OLD STATUS ON ILLEGAL JUMP TRAP
 7582 033160  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7583 033164  016767  144620  147620          MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7584 033172  012767  033214  144610          MOV     #MILOB,10               ;SETUP NEW ILLEGAL VECTOR
 7585 033200  005067  144572                  CLR     PS                      ;CLEAR PRIORITY AND COND C
 7586 033204  000257                          CCC
 7587 033206  005001                          CLR     R1
 7588 033210  000101                          JMP     R1
 7589 033212  104001                  MILOA:  ERROR   +1              ;CPU ERROR
 7590                                                                 ;DIDNT TAKE CORRECT TRAP
 7591 033214  026727  145556  000004  MILOB:  CMP     STBOT-2,#4              ;VERIFY PSW ON STACK
 7592 033222  001401                          BEQ     MILOC                   ;BRANCH IF CORRECT STATUS
 7593 033224  104001                          ERROR   +1              ;CPU ERROR
 7594                                                                 ;BAD STATUS ON STACK
 7595 033226  012706  001000          MILOC:  MOV     #STBOT,R6               ;SETUP STACK
 7596 033232  012767  033254  144550          MOV     #MILOE,10               ;SET UP TRAP VECTOR
 7597 033240  012767  000357  144530          MOV     #357,PS                 ;SET PRIORITY
 7598 033246  000277                          SCC                             ;SET CONDITION CODES
 7599 033250  000101                          JMP     R1
```

```
********** DOUBLE OPERAND TESTS ******************

7600 033252  104001                    MILOD:  ERROR   +1              ;CPU ERROR
7601                                                                    ;DIDNT TAKE CORRECT TRAP
7602 033254  026727  145516  000357     MILOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7603 033262  001401                            BEQ     MILOF                   ;BRANCH IF GOOD
7604 033264  104001                            ERROR   +1              ;CPU ERROR
7605                                                                    ;OLD PSW INCORRECT
7606 033266                            MILOF:
7607 033266  016767  147520  144514            MOV     SLOC00,10               ;RESTORE VECTOR
7608 033274  012706  001000                    MOV     #STBOT,R6
7609
7611                                   ;
7612                                   ;
7614 033300                            MIALL:
7615
7616                                   ;       TEST ILLEGAL JSR INSTRUCTION TRAP
7617 033300  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7618 033304  016767  144500  147500            MOV     10,SLOC00               ;SAVE OLD  VECTOR
7619 033312  012767  033326  144470            MOV     #MIALLB,10              ;SETUP NEW ILLEGAL VECTOR
7620 033320  005003                            CLR     R3
7621 033322  004303                            JSR     R3,R3
7622 033324  104001                    MIALLA: ERROR   +1              ;CPU ERROR
7623                                                                    ;DIDNT TAKE CORRECT TRAP
7624 033326  022706  000774            MIALLB: CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7625 033332  001401                            BEQ     MIALLD                  ;BRANCH IF GOOD
7626 033334  104001                            ERROR   +1              ;CPU ERROR
7627                                                                    ;BAD PC ON STACK
7628 033336  021627  033324            MIALLD: CMP     (R6),#MIALLA            ;VERFY PROPER PC ON STACK
7629 033342  001401                            BEQ     MIALLF                  ;BRANCH IF GOOD
7630 033344  104001                            ERROR   +1              ;CPU ERROR
7631                                                                    ;INCORRECT PC ON STACK
7632 033346  016767  147440  144434  MIALLF: MOV     SLOC00,10               ;RESTORE VECTOR
7633 033354  012706  001000                    MOV     #STBOT,R6
7634
7636                                   ;
7637                                   ;
7639 033360                            MJSI:
7640
7641                                   ;       TEST OLD STATUS ON ILLEGAL JSR TRAP
7642 033360  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7643 033364  016767  144420  147420            MOV     10,SLOC00               ;SAVE OLD  VECTOR
7644 033372  012767  033414  144410            MOV     #MJSIB,10               ;SETUP NEW VECTOR
7645 033400  005067  144372                    CLR     PS                      ;CLEAR PRIORITY AND COND C
7646 033404  000257                            CCC
7647 033406  005003                            CLR     R3
7648 033410  004303                            JSR     R3,R3
7649 033412  104001                    MJSIA:  ERROR   +1              ;CPU ERROR
7650                                                                    ;DIDNT TAKE CORRECT TRAP
7651 033414  026727  145356  000004    MJSIB:  CMP     STBOT-2,#4              ;VERIFY PSW ON STACK
7652 033422  001401                            BEQ     MJSIC                   ;BRANCH IF CORRECT STATUS
7653 033424  104001                            ERROR   +1              ;CPU ERROR
7654                                                                    ;BAD STATUS ON STACK
7655 033426  012706  001000            MJSIC:  MOV     #STBOT,R6               ;SETUP STACK
7656 033432  012767  033454  144350            MOV     #MJSIE,10               ;SET UP TRAP VECTOR
7657 033440  012767  000357  144330            MOV     #357,PS                 ;SET PRIORITY
7658 033446  000277                            SCC                             ;SET CONDITION CODES
7659 033450  004303                            JSR     R3,R3
7660 033452  104001                    MJSID:  ERROR   +1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS ******************

 7661                                                          ;DIDNT TAKE CORRECT TRAP
 7662 033454  026727  145316  000357  MJSIE:  CMP     STBOT-2,#357          ;VERIFY OLD PSW ON STACK
 7663 033462  001401                          BEQ     MJSIF                 ;BRANCH IF GOOD
 7664 033464  104001                          ERROR   .+1                   ;CPU ERROR
 7665                                                          ;OLD PSW INCORRECT
 7666 033466                          MJSIF:
 7667 033466  016767  147320  144314          MOV     SLOC00,10             ;RESTORE VECTOR
 7668 033474  012706  001000                  MOV     #STBOT,R6
 7669
 7671                                 ;
 7672                                 ;
 7674 033500                          IOXXX:
 7675                                 ;       I/O TIME OUT TEST
 7676 033500  005067  144262                  CLR     CPEREG                ;CLEAR CPU ERROR REGISTER
 7677 033504  016767  144274  147300          MOV     4,SLOC00              ;SAVE VECTOR
 7678 033512  012767  033534  144264          MOV     #2$,4                 ;SET UP VECTOR TO HANDLE NXM
 7679 033520  012767  030000  144250          MOV     #30000,PS             ;INIT THE PSW TO A KNOWN STATE
 7680 033526  005737  177700                  TST     @#177700              ;TRY TO ACCESS HARDWARE ADDRESS
 7681                                                          ;FOR GENERAL PURPOSE REG 0. THIS
 7682                                                          ;IS NOT IMPLEMENTED ON KDJ11
 7683                                                          ;SHOULD CAUSE TIME OUT.
 7684 033532  104001                  1$:     ERROR   .+1                   ;CPU ERROR
 7685 033534  022767  000020  144224  2$:     CMP     #BIT04,CPEREG         ;IS CPU ERROR REGISTER CORRECT?
 7686 033542  001401                          BEQ     3$
 7687 033544  104001                          ERROR   .+1                   ;CPU ERROR
 7688 033546  022627  033532          3$:     CMP     (SP)+,#1$             ;CHECK THAT STACK CONTAINS CORRECT ADDR.
 7689 033552  001401                          BEQ     4$                    ;
 7690 033554  104001                          ERROR   .+1                   ;CPU ERROR
 7691 033556  022627  030000          4$:     CMP     (SP)+,#30000          ;IS THE PSW OK?
 7692 033562  001401                          BEQ     5$                    ;
 7693 033564  104001                          ERROR   .+1                   ;CPU ERROR
 7694 033566  005067  144174          5$:     CLR     CPEREG                ;CLEAR THE CPU ERROR REGISTER
 7695 033572  016767  147214  144204          MOV     SLOC00,4              ;RESTORE VECTOR
 7696
 7699 033600                          ODDXX:
 7708
 7709                                 ;       ODD ADDRESS/ILLEGAL INST FETCH TRAP TEST
 7710                                 ;;**************************************************************************
                                      ;THIS PROGRAM GENERATES AN ODD ADDRESS IN THE PC. THE KDJ11 SHOULD
                                      ;TRAP THROUGH ADDR 4
 7711 033600  005067  144162                  CLR     CPEREG                ;INIT THE CPU ERROR REG
 7712 033604  016767  144174  147200          MOV     4,SLOC00              ;SAVE VECTOR
 7713 033612  012767  033646  144164          MOV     #2$,4                 ;SET UP VECTOR TO HANDLE ODD ADDR TRAP
 7714 033620  016767  144162  147166          MOV     6,SLOC01              ;SAVE VECTOR
 7715 033626  005067  144154                  CLR     6                     ;INIT VECTOR
 7716 033632  012746  030000                  MOV     #30000,-(SP)          ;PUSH A KNOWN PSW ON THE STACK
 7717 033636  012746  033645                  MOV     #1$+1,-(SP)           ;PUSH AN ODD NUMBER ON THE STACK
 7718 033642  000002                          RTI                           ;POP ODD ADDRESS OFF STACK INTO PC
 7719                                                          ;SHOULD TRAP HERE
 7720 033644  104001                  1$:     ERROR   .+1                   ;CPU ERROR
 7721 033646  022767  000100  144112  2$:     CMP     #BIT06,CPEREG         ;IS CPU ERROR REGISTER CORRECT?
 7722 033654  001401                          BEQ     3$
 7723 033656  104001                          ERROR   .+1                   ;CPU ERROR
 7724 033660  022726  033645          3$:     CMP     #1$+1,(SP)+           ;IS STACK CONTENTS CORRECT?
 7725 033664  001401                          BEQ     4$                    ;
 7726 033666  104001                          ERROR   .+1                   ;CPU ERROR
 7727 033670  022726  030000          4$:     CMP     #30000,(SP)+          ;IS PSW CORRECT ON STACK?
```

```
********** DOUBLE OPERAND TESTS **********

7728 033674  001401                        BEQ     5$                      ;
7729 033676  104001                        ERROR   .1                ;CPU ERROR
7730 033700  005067  144062          5$:   CLR     CPEREG                  ;CLEAR CPU ERROR REG
7731                                       ;
7732                                       ;NOW WE'LL TRY TO FETCH AN INSTRUCTION FROM AN INTERNAL REGISTER.
7733                                       ;THIS SHOULD CAUSE A TRAP TO ADDR 4 AND SET BIT 6 IN THE CPU
7734                                       ;ERROR REGISTER.
7735                                       ;
7736 033704  012767  033732  144072        MOV     #7$,4                   ;LOAD VECTOR WITH TRAP HANDLER ADDR
7737 033712  012767  030340  144066        MOV     #30340,6                ;LOAD VEC WITH PSW VALUE ON TRAP
7738 033720  005067  144052                CLR     PS                      ;CLEAR THE PSW
7739 033724  000167  144046                JMP     PS                      ;********TEST INSTRUCTION*********
7740                                                                       ;TRY INSTRUCTION FETCH FROM INTERNAL
7741                                                                       ;REGISTER-- SHOULD TRAP VIA ADDR 4
7742 033730  104001                  6$:   ERROR   .1                ;CPU ERROR
7743 033732  016701  144040          7$:   MOV     PS,R1                   ;SAVE CONTENTS OF PSW IN R1
7744 033736  022767  000100  144022        CMP     #BIT06,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
7745 033744  001401                        BEQ     8$                      ;
7746 033746  104001                        ERROR   .1                ;CPU ERROR
7747 033750  022726  177776          8$:   CMP     #PS,(SP)+               ;IS STACK CONTENTS CORRECT?
7748 033754  001401                        BEQ     9$                      ;
7749 033756  104001                        ERROR   .1                ;CPU ERROR
7750 033760  022726  000000          9$:   CMP     #0,(SP)+                ;IS STACK CONTENTS CORRECT?
7751 033764  001401                        BEQ     10$                     ;
7752 033766  104001                        ERROR   .1                ;CPU ERROR
7753 033770  022701  000340          10$:  CMP     #340,R1                 ;WAS PSW LOADED PROPERLY ON TRAP?
7754 033774  001401                        BEQ     11$                     ;
7755 033776  104001                        ERROR   .1                ;CPU ERROR
7756 034000  005067  143762          11$:  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7757 034004  016767  147002  143772        MOV     SLOC00,4                ;RESTORE VECTOR
7758 034012  016767  146776  143766        MOV     SLOC01,6                ;       "
7759
7762 034020                          RXXX:
7763
7764                                       ;       RED ZONE TRAP TEST
7765 034020  013767  000004  146764        MOV     @#4,SLOC00              ;SAVE VECTOR
7766 034026  012737  034056  000004        MOV     #2$,@#4                 ;SET UP VECTOR
7767 034034  012706  000777                MOV     #777,R6                 ;SET UP THE STACK WITH ODD ADDRESS.
7768 034040  005067  143722                CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
7769 034044  005067  143726                CLR     PSW                     ;CLEAR THE PSW
7770 034050  005737  177700                TST     @#177700                ;ACCESS NON-EXISTANT I/O ADDRESS
7771 034054  104001                  1$:   ERROR   .1                ;CPU ERROR
7772 034056  022706  000000          2$:   CMP     #0,SP                   ;IS R6 CORRECT?
7773 034062  001401                        BEQ     3$                      ;BRANCH IF YES
7774 034064  104001                        ERROR   .1                ;CPU ERROR
7775 034066  022726  034054          3$:   CMP     #1$,(SP)+               ;IS DATA AT ADDR 0 CORRECT?
7776 034072  001401                        BEQ     4$                      ;BRANCH IF YES
7777 034074  104001                        ERROR   .1                ;CPU ERROR
7778 034076  022716  000000          4$:   CMP     #0,(SP)                 ;IS PSW DATA IN ADDR 2 CORRECT?
7779 034102  001401                        BEQ     5$                      ;BRANCH IF YES
7780 034104  104001                        ERROR   .1                ;CPU ERROR
7781 034106  022767  000124  143652  5$:   CMP     #124,CPEREG             ;IS CPU ERROR REGISTER CORRECT?
7782 034114  001401                        BEQ     6$                      ;BRANCH IF YES
7783 034116  104001                        ERROR   .1                ;CPU ERROR
7784 034120  005067  143642          6$:   CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7785 034124  012706  001000                MOV     #STBOT,SP               ;RESTORE STACK
7786 034130  016737  146656  000004        MOV     SLOC00,@#4              ;RESTORE VECTOR
```

```
********** DOUBLE OPERAND TESTS ******************

 7787 034136  005037  000000                CLR     @#0                     ;RESTORE ADDR 0
 7788 034142  005037  000002                CLR     @#2                     ;RESTORE ADDR 2
 7789
 7791
 7793
 7794 034146                        PIRXXX:
 7795                               ;       TEST PIRQ REGISTER RESPONSE
 7796 034146  016767  143632  146636        MOV     4,SLOC00                ;SAVE CONTENTS OF VECTORS
 7797 034154  012767  034202  143622        MOV     #PIRQNX,4               ;SETUP INTERRUPT VECTOR
 7798 034162  005067  143600                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7799 034166  005737  177772                TST     @#PIRQ                  ;LOOK FOR REPLY FROM PIRQ
 7800 034172  016767  146614  143604        MOV     SLOC00,4                ;RESTORE VECTORS
 7801 034200  000401                        BR      PIRTEX                  ;IF IT RESPONDS, CONTINUE TESTING.
 7802                                                                       ;ERROR! NO RESPONSE FROM PIRQ REG
 7803 034202  104001                PIRQNX: ERROR   +1              ;CPU ERROR
 7804 034204                        PIRTEX:
 7805
 7806 034204                        PIR1:
 7807                               ;       TEST PIRQ REGISTER DATA
 7808 034204  013767  000240  146600        MOV     @#PIRQVEC,SLOC00        ;SAVE PIRQ VECTORS
 7809 034212  013767  000242  146574        MOV     @#PIRQVEC+2,SLOC01      ;       "
 7810 034220  012737  034430  000240        MOV     #UNXPIR,@#PIRQVEC       ;SET UP PIRQ VECTOR FOR UNEXPECTED INTERRUPT
 7811 034226  012737  000340  000242        MOV     #PR7,@#PIRQVEC+2        ;
 7812 034234  012703  001000                MOV     #1000,R3                ;PUT 1000 IN R3: START TESTING
 7813                                                                       ;BITS IN PIRQ REG BY FLOATING
 7814                                                                       ;A BIT THROUGH BITS 9-15.
 7815 034240  012704  034304                MOV     #PIRTBL,R4              ;SET UP R4 AS A POINTER TO EXPECTED
 7816                                                                       ;ENCODED PRIORITY LEVELS IN PIRTBL
 7817 034244  000237                1$:     SPL     7                       ;DON'T ALLOW INTERRUPTS.
 7818 034246  005037  177772                CLR     @#PIRQ                  ;CLEAR OUT THE PIRQ
 7819 034252  023724  177772                CMP     @#PIRQ,(R4)+            ;IS PIRQ OK??
 7820 034256  001401                        BEQ     2$                      ;BRANCH IF OK
 7821                                                                       ;ERROR; PIRQ REG WAS NOT CLEAR
 7822 034260  104001                        ERROR   +1              ;CPU ERROR
 7823 034262  010337  177772        2$:     MOV     R3,@#PIRQ               ;SET A BIT IN PIRQ REGISTER
 7824 034266  023724  177772                CMP     @#PIRQ,(R4)+            ;COMPARE THE ENCODED PRIORITY BITS
 7825                                                                       ;WITH DATA IN THE TABLE (PIRTBL)
 7826 034272  001401                        BEQ     3$                      ;BRANCH IF ITS OK
 7827                                                                       ;ERROR; ENCODED PRIORITY LEVELS ARE
 7828                                                                       ;NOT CORRECT
 7829 034274  104001                        ERROR   +1              ;CPU ERROR
 7830 034276  006303                3$:     ASL     R3                      ;FLOAT A "1" THRU BITS 9-15
 7831 034300  103370                        BCC     2$                      ;IF CARRY BIT ISN'T SET, DO AGAIN.
 7832 034302  000410                        BR      EXPIR1                  ;GO TO EXIT TEST.
 7833
 7834 034304  000000                PIRTBL: .WORD   0
 7835 034306  001042                        .WORD   1042
 7836 034310  002104                        .WORD   2104
 7837 034312  004146                        .WORD   4146
 7838 034314  010210                        .WORD   10210
 7839 034316  020252                        .WORD   20252
 7840 034320  040314                        .WORD   40314
 7841 034322  100356                        .WORD   100356
 7842
 7843 034324                        EXPIR1:
 7844
 7854
```

********** DOUBLE OPERAND TESTS ******************

```
7855 034324                            PIR2:
7856                                   :       TEST PIRQ REGISTER LEVEL ENCODING
7857                                   ;;**************************************************************************
                                       ;THIS TEST IS TO CHECK THAT THE HIGHEST PRIORITY LEVEL SET IN THE
                                       ;PROGRAM INTERRUPT REQUEST BITS IS REFLECTED IN THE ENCODED PROGRAM
                                       ;INTERRUPT ACTIVE BITS.
7858 034324  000237                            SPL     7                       ;SHUT OFF INTERRUPTS
7859 034326  005037  177772                    CLR     @#PIRQ                  ;CLEAR PIRQ REGISTER
7860 034332  012767  000001  146470            MOV     #1,FLAG                 ;SETUP END OF LOOP SIGNAL
7861 034340  012703  177000                    MOV     #177000,R3              ;SET UP DESIRED PATTERN IN R3
7862 034344  012704  034410                    MOV     #PITBL1,R4              ;SETUP R4 AS A TABLE POINTER
7863 034350  010337  177772            1$:     MOV     R3,@#PIRQ               ;SET PATERN IN PIRQ REGISTER
7864
7865 034354  023724  177772                    CMP     @#PIRQ,(R4)+            ;COMPARE PATTERN IN PIRQ WITH PATTERN IN
7866                                                                           ;EXPECTED PATTERN TABLE.
7867 034360  001012                            BNE     2$                      ;GO TO ERROR IF NOT THE SAME
7868 034362  005737  003030                    TST     @#FLAG                  ;IS THIS THE LAST TIME THROUGH??
7869 034366  001421                            BEQ     PIR2EX                  ;YES, IF FLAG IS ZERO
7870 034370  006003                            ROR     R3                      ;SHIFT PATTERN TO RIGHT FOR NEXT TEST
7871 034372  042703  000777                    BIC     #777,R3                 ;STRIP OFF BITS  8-0
7872 034376  001364                            BNE     1$                      ;IF R3 IS NOT = 0 GO DO THE NEXT PATTERN
7873 034400  005037  003030                    CLR     @#FLAG                  ;CLR THE FLAG TO INDICATE LAST
7874                                                                           ;TIME THROUGH THE LOOP
7875 034404  000761                            BR      1$                      ;GO THROUGH LOOP ONCE MORE
7876                                                                           ;ERROR; PATTERNS WERE NOT THE SAME
7877 034406  104001                    2$:     ERROR   +1                      ;CPU ERROR
7878
7879 034410  177356  077314  037252    PITBL1: .WORD   177356,77314,37252,17210,7146,3104,1042,0
     034416  017210  007146  003104
     034424  001042  000000
7880
7881 034430                            UNXPIR:                                 ;UNEXPECTED PIRQ INTERRUPT
7882 034430  104001                            ERROR   +1                      ;CPU ERROR
7883
7884 034432                            PIR2EX:
7892
7893 034432                            PIR3:
7894                                   ;       TEST PIRQ INTERRUPTS
7895                                   ;;**************************************************************************
                                       ;THIS TEST CHECKS THAT EACH PROGRAM INTERRUPT OCCURS PROPERLY
7896 034432  012703  001000                    MOV     #1000,R3                ;SETUP R3 AS A WORKING REGISTER
7897 034436  012737  034502  000240            MOV     #PIRRTN,@#PIRQVEC       ;SET UP INTERRUPT ROUTINE AT PIR VECTOR
7898 034444  012737  000340  000242            MOV     #340,@#PIRQVEC+2        ;       "       "       "       "
7899 034452  005004                            CLR     R4                      ;INITIALIZE R4 AS EXPECTED DATA HOLDER.
7900 034454  005724                    PI1:    TST     (R4)+                   ;INCREMENT R4 BY 2
7901 034456  050337  177772                    BIS     R3,@#PIRQ               ;SET PIRQ TO INTERRUPT
7902 034462  000230                            SPL     0                       ;ENABLE INTERRUPT TO OCCUR
7903                                                                           ;ERROR! INTERRUPT DID NOT OCCUR
7904 034464  104001                            ERROR   +1                      ;CPU ERROR
7905                                                                           ;ERROR! INTERRUPT OCCURRED IN WRONG SEQUENCE
7906 034466  104001                    PI2:    ERROR   +1                      ;CPU ERROR
7907 034470  062706  000004            PI3:    ADD     #4,SP                   ;CLEAN UP THE STACK
7908 034474  006303                            ASL     R3                      ;SHIFT R3 TO LEFT FOR NEXT INTERRUPT
7909 034476  103366                            BCC     PI1                     ;END IF CARRY BIT IS SET
7910 034500  000412                            BR      PIR3EX                  ;ON TO THE NEXT TEST
7911
7912 034502  013705  177772            PIRRTN: MOV     @#PIRQ,R5               ;MOVE THE CONTENTS OF PIRQ REG TO R5
```

```
********** DOUBLE OPERAND TESTS **********

7913 034506  005037  177772                CLR     @#PIRQ                  ;KILL PIRQ INTERRUPT.
7914 034512  042705  177761                BIC     #177761,R5              ;MASK OFF ALL BITS EXCEPT 1-4.
7915 034516  020405                        CMP     R4,R5                   ;DID THE CORRECT LEVEL INTERRUPT OCCUR?
7916 034520  001362                        BNE     PI2                     ;IF NOT; GO TO ERROR.
7917 034522  000167  177742                JMP     PI3                     ;RETURN FROM SUCCESSFUL INTERRUPT,GET
7918                                                                        ;READY FOR THE NEXT ONE.
7919
7920 034526                        PIR3EX.
7921
7930
7931 034526                        PIR4:
7932                               ;       TEST PIRQ VS PSW INTERRUPT LEVEL
7933                               ;;**************************************************************
                                   ;THIS TEST IS TO ENSURE THAT PIRQ CANNOT INTERRUPT WHEN PSW INTERRUPT
                                   ;LEVEL IS SET TO BLOCK THEM OUT.
7934 034526  005037  177772                CLR     @#PIRQ                  ;CLEAR THE PIRQ
7935 034532  005037  177776                CLR     @#PSW                   ;CLEAR THE PSW.
7936 034536  000237                        SPL     7                       ;BLOCK INTERRUPTS FROM BEING SERVICED.
7937 034540  012703  001000                MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
7938 034544  012737  034576  000240        MOV     #2$,@#PIRQVEC           ;SETUP INTERRUPT VECTORS
7939 034552  012737  000340  000242        MOV     #340,@#PIRQVEC+2        ;
7940 034560  010337  177772        1$:     MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
7941 034564  006303                        ASL     R3                      ;SHIFT A "1" LEFT TO INCREASE INTERRUPT
7942                                                                        ;PRIORITY LEVEL.
7943 034566  103374                        BCC     1$                      ;IF C BIT NOT SET THEN DO IT AGAIN.
7944 034570  005037  177772                CLR     @#PIRQ                  ;ELSE CLEAR THE PIRQ
7945 034574  000403                        BR      3$                      ;EXIT TEST.
7946
7947 034576  005037  177772        2$:     CLR     @#PIRQ                  ;STOP PIRQ FROM INTERRUPTING.
7948                                                                        ;ERROR! NO INTERRUPTS SHOULD OCCUR.
7949 034602  104001                        ERROR   .1              ;CPU ERROR
7950 034604                        3$:
7951
7965
7966 034604                        PIR5:
7967                               ;       TEST PIRQ INTERRUPTS OCCUR AT PROPER LEVEL
7968                               ;;**************************************************************
                                   ;THIS TEST ENSURES THAT INTERRUPTS OCCUR AT THE PROPER LEVEL
                                   ;THE PRIORITY LEVEL IS INITIALLY SET TO PRI6. THE PIRQ THEN STARTS INTERRUPTING
                                   ;AT PRI1. NO INTERRUPTS SHOULD OCCUR UNTIL THE INTERRUPTING LEVEL EXCEEDS THE
                                   ;PRIORITY LEVEL IN THE PSW. AFTER EACH LEVEL OF INTERRUPT HAS BEEN TRIED; THE
                                   ;PSW PRIORITY LEVEL IS LOWERED ONE NOTCH, AND THE CYCLE REPEATS UNTIL PSW
                                   ;PRIORITY LEVEL 0 IS REACHED. INTERRUPTS AT PSW PRIORITY LEVEL 0 HAVE BEEN DONE
                                   ;PREVIOUSLY.
7969 034604  005037  177772                CLR     @#PIRQ                  ;CLR THE PIRQ
7970 034610  012737  034722  000240        MOV     #10$,@#PIRQVEC          ;SET UP INTERRUPT VECTORS
7971 034616  012737  000340  000242        MOV     #PR7,@#PIRQVEC+2        ;
7972 034624  012705  035000                MOV     #PITBL2,R5              ;SETUP R5 AS A TABLE POINTER.
7973 034630  012737  000006  111460        MOV     #6,@#DCOUNT             ;INIT LOCATION DCOUNT:
7974 034636  005004                1$:     CLR     R4                      ;INITIALIZE R4 AS A COUNTER.
7975 034640  012703  001000                MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
7976 034644  012567  143126                MOV     (R5)+,PS                ;MOVE PRIORITY LEVEL TO PSW
7977 034650  005724                3$:     TST     (R4)+                   ;INCREMENT R4 BY 2
7978 034652  010337  177772                MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
7979                               ;*******INTERRUPTS WILL HAPPEN HERE*********
7980 034656  013701  177776                MOV     @#PS,R1                 ;SAVE PS IN R1          !ONLY EXECUTED
7981 034662  013702  177772                MOV     @#PIRQ,R2               ;SAVE PIRQ IN R2        !   IF NO
```

```
034666  042701  177437                  BIC     #177437,R1              ;CLEAR EXTRANEOUS BITS  ! INTERRUPT
034672  042702  177437                  BIC     #177437,R2              ;                       !    HAS
034676  020102                          CMP     R1,R2                   ;R1 SHOULD BE >= R2     !  OCCURRED
034700  002001                          BGE     4$                      ;IF IT IS GO ON         !-------
                                                                        ;ERROR! SHOULD HAVE INTERRUPTED.
034702  104001                          ERROR   +1                ;CPU ERROR
034704  006303                  4$:     ASL     R3                      ;SHIFT A "1" LEFT UNTIL INTERRUPT LEVEL
                                                                        ;IS REACHED.
034706  103360                          BCC     3$                      ;BRANCH BACK IF CARRY IS NOT SET.
034710  005337  111460                  DEC     @#DCOUNT                ;LOWER INTERRUPT PRIORITY LEVEL
034714  001350                          BNE     1$                      ;IF DCOUNT= 0 , WE'RE DONE.
034716  000167  000072                  JMP     PIR5EX                  ;JUMP TO END OF THIS TEST.

                                ;PIRQ INTERRUPT SERVICE ROUTINE
                                ;

034722  013746  177772          10$:    MOV     @#PIRQ,-(SP)            ;SAVE PIRQ DATA ON STACK
034726  005037  177772                  CLR     @#PIRQ                  ;SHUT OFF PIRQ INTERRUPTS
034732  016601  000004                  MOV     4(SP),R1                ;GET OLD PSW. SAVE IN R1.
034736  011602                          MOV     (SP),R2                 ;GET PIRQ FROM STACK.
034740  042702  177437                  BIC     #177437,R2              ;CLEAR UNWANTED BITS FROM R2
034744  042701  177437                  BIC     #177437,R1              ;CLEAR UNWANTED BITS FROM R1
034750  020102                          CMP     R1,R2                   ;R2 SHOULD BE > R1.
034752  100401                          BMI     20$                     ;GO CHECK SEQUENCE OF INTERRUPT.
                                                                        ;ERROR! PRIORITY OF INTERRUPT WAS NOT
                                                                        ;HIGH ENOUGH. SHOULDN'T HAVE OCCURRED.
034754  104001                          ERROR   +1                ;CPU ERROR
034756  012602                  20$:    MOV     (SP)+,R2                ;POP OLD PIRQ OFF THE STACK.
034760  042702  177761                  BIC     #177761,R2              ;CLEAR OFF EXTRANEOUS BITS.
034764  020402                          CMP     R4,R2                   ;SHOULD BE EQUAL.
034766  001401                          BEQ     21$                     ;IF THEY ARE EQUAL, CLEAN UP STACK AND
                                                                        ;GET READY FOR THE NEXT INTERRUPT
                                                                        ;ELSE
                                                                        ;ERROR! INTERRUPT OCCURRED OUT OF SEQUENCE.
034770  104001                          ERROR   +1                ;CPU ERROR
034772  012716  034704          21$:    MOV     #4$,(SP)                ;PUT RETURN ADDRESS ON THE STACK.
034776  000002                          RTI                             ;RETURN FROM INTERRUPT. RESTORE PSW.

035000  000300                  PITBL2: .WORD   300               ;PRIORITY LEVEL 6
035002  000240                          .WORD   240               ;PRIORITY LEVEL 5
035004  000200                          .WORD   200               ;PRIORITY LEVEL 4
035006  000140                          .WORD   140               ;PRIORITY LEVEL 3
035010  000100                          .WORD   100               ;PRIORITY LEVEL 2
035012  000040                          .WORD   40                ;PRIORITY LEVEL 1

035014                          PIR5EX:


035014                          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.
```

```
********** DOUBLE OPERAND TESTS **********

                                        ;AFTER A CORRECT INTERRUPT, AN RTI IS EXECUTED, AND THE PSW PRIORITY LEVEL
                                        ;RETRUNS TO ZERO ALLOWING THE NEXT LOWER INTERRUPT TO OCCUR.
 8047 035014  005037  177772                    CLR     @#PIRQ                  ;CLEAR OUT THE PIRQ
 8048 035020  000237                            SPL     7                       ;NO INTERRUPTS WILL BE SERVICED.
 8049 035022  012737  035062  000240            MOV     #10$,@#PIRQVEC          ;SET UP VECTORS
 8050 035030  012737  000340  000242            MOV     #PR7,@#PIRQVEC+2        ;
 8051 035036  012703  177000                    MOV     #177000,R3              ;SETUP DESIRED PATTERN IN R3
 8052 035042  010337  177772                    MOV     R3,@#PIRQ               ;MOVE PATTERN IN R3 TO PIRQ.
 8053 035046  012704  000016                    MOV     #16,R4                  ;PRELOAD R4 AS A DECREMENTING COUNTER.
 8054 035052  000230                            SPL     0                       ;LET THE PIRQ INTERRUPT
 8055                                   ;*********INTERRUPTS OCCUR HERE!!!*********************
 8056 035054  005704                            TST     R4                      ;IS R4 ZERO? IT SHOULD BE. THIS
 8057                                                                           ;INSTRUCTION SHOULDN'T BE EXECUTED
 8058                                                                           ;UNTIL ALL INTERRUPTS HAVE OCCURRED.
 8059 035056  001417                            BEQ     PIR6EX                  ;GO TO NEXT TEST.
 8060                                                                           ;ERROR! ALL INTERRUPTS DID NOT OCCUR.
 8061 035060  104001                            ERROR   +1              ;CPU ERROR
 8062 035062  013702  177772            10$:    MOV     @#PIRQ,R2               ;SAVE THE PIRQ IN R2
 8063 035066  042702  177761                    BIC     #177761,R2              ;STRIP OFF EXTRANEOUS BITS.
 8064 035072  020402                            CMP     R4,R2                   ;SHOULD BE EQUAL.
 8065 035074  001401                            BEQ     15$                     ;SETUP FOR NEXT INTERRUPT.
 8066                                                                           ;ERROR! INTERRUPT WAS SERVICED OUT OF ORDER
 8067 035076  104001                            ERROR   +1              ;CPU ERROR
 8068 035100  005744                    15$:    TST     -(R4)                   ;DECREMENT R4 BY 2
 8069 035102  006003                            ROR     R3                      ;CHANGE PATTERN IN R3; LOWER INTERRUPT PRIORITY
 8070 035104  042703  000777                    BIC     #777,R3                 ;STRIP OFF UNNEEDED BITS
 8071 035110  010337  177772                    MOV     R3,@#PIRQ               ;LOWER INTERRUPT PRIORITY LEVEL.
 8072 035114  000002                            RTI                             ;
 8073
 8074 035116  016737  145670  000240  PIR6EX: MOV     SLOC00,@#PIRQVEC        ;RESTORE VECTORS
 8075 035124  016737  145664  000242            MOV     SLOC01,@#PIRQVEC+2              ;       "
 8076                                                   ;
 8078
 8081                                   .SBTTL  MEMORY MANAGEMENT TESTS
 8082                                   ;*******************************************************************************
 8083                                   ;*******************************************************************************
 8084                                   ;                       BEGIN MMU TESTING
 8085                                   ;*******************************************************************************
 8086                                   ;*******************************************************************************
 8088 035132                            TSMMU1:
 8089                                   ;       STATUS REGISTER TEST
 8090 035132  005067  142630                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 8091 035136  005037  177572                    CLR     @#177572                ;TURN MMU OFF
 8092 035142  005037  003030                    CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
 8093 035146  013746  000004                    MOV     @#4,-(SP)               ;SAVE OLD VECTOR
 8094 035152  012737  135234  000004            MOV     #ADDTRP,@#4             ;SETUP NEW VECTOR
 8095 035160  005005                            CLR     R5                      ;CLEAR FLAG
 8096 035162  013701  177572                    MOV     @#177572,R1             ; TEST MMR0
 8097 035166  013701  177574                    MOV     @#177574,R1             ; TEST MMR1
 8098 035172  013701  177576                    MOV     @#177576,R1             ; TEST MMR2
 8099 035176  013701  172516                    MOV     @#172516,R1             ; TEST MMR3
 8100 035202  012637  000004                    MOV     (SP)+,@#4               ;RESTORE VECTOR
 8101 035206  020527  000000                    CMP     R5,#0                   ;DID WE TRAP
 8102 035212  001401                            BEQ     1$                      ;NO, THEN BRANCH
 8103 035214  104002                            ERROR   +2              ;MMU ERROR
                                                                        ;YES, GO TO ERROR
 8104
 8105 035216                            1$:
```

MEMORY MANAGEMENT TESTS

```
8108 035216                                    TSMMU2:
8109                                           ;       ADDRESS TEST OF PARS,PDRS, AND FP REGS
8110 035216  005067  142544                             CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
8111 035222  005037  177572                             CLR     @#177572                ;MMU OFF
8112 035226  005037  003030                             CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
8113 035232  013746  000244                             MOV     @#244,-(SP)             ;SAVE FP VECTOR
8114 035236  013746  000246                             MOV     @#246,-(SP)
8115 035242  013746  000004                             MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
8116 035246  012737  000246  000244                     MOV     #246,@#244              ;SETUP NEW FP VECTOR
8117 035254  012737  000002  000246                     MOV     #2,@#246                ;
8118 035262  012737  135234  000004                     MOV     #ADDTRP,@#4             ;SETUP NEW TIME OUT VECTOR
8119 035270  005005                                     CLR     R5                      ;CLEAR TIMEOUT FLAG
8120 035272  012700  172200                             MOV     #172200,R0              ;LOAD ALL PARS AND PDRS WITH ZERO
8121 035276  005020                             1$:     CLR     (R0)+                   ;
8122 035300  020027  172400                             CMP     R0,#172400              ;
8123 035304  001374                                     BNE     1$                      ;
8124 035306  012700  177600                             MOV     #177600,R0              ;
8125 035312  005020                             2$:     CLR     (R0)+                   ;
8126 035314  020027  177700                             CMP     R0,#177700              ;
8127 035320  001374                                     BNE     2$                      ;
8128 035322  170127  000200                             LDFPS   #200                    ;
8129 035326  012700  003052                             MOV     #FLOAT,R0               ;LOAD AC0-AC5 WITH 0
8130 035332  005020                                     CLR     (R0)+                   ;
8131 035334  005020                                     CLR     (R0)+                   ;
8132 035336  005020                                     CLR     (R0)+                   ;
8133 035340  005020                                     CLR     (R0)+                   ;
8134 035342  012700  003052                             MOV     #FLOAT,R0               ;
8135 035346  172410                                     LDD     (R0),AC0                ;
8136 035350  172510                                     LDD     (R0),AC1                ;
8137 035352  172610                                     LDD     (R0),AC2                ;
8138 035354  172710                                     LDD     (R0),AC3                ;
8139 035356  174004                                     STD     AC0,AC4                 ;
8140 035360  174005                                     STD     AC0,AC5                 ;
8141 035362  174500                             3$:     DIVD    AC0,AC1                 ;LOAD FEC WITH 4 AND FEA WITH @3$
8142 035364  170337  003062                             STST    @#FLO                   ;CHECK FEC FOR 4 AND FEA FOR @3$
8143 035370  012704  003062                             MOV     #FLO,R4                 ;
8144 035374  022427  000004                             CMP     (R4)+,#4                ;
8145 035400  001401                                     BEQ     21$                     ;
8146 035402  104002                                     ERROR   +2              ;MMU ERROR
8147                                                                            ;
8148 035404  021427  035362                     21$:    CMP     (R4),@3$                ;
8149 035410  001401                                     BEQ     22$                     ;
8150 035412  104002                                     ERROR   +2              ;MMU ERROR
8151                                                                            ;
8152 035414  012704  172200                     22$:    MOV     #172200,R4              ;CHECK EACH PAR, PDR FOR 0 THEN
8153 035420  012701  000001                             MOV     #1,R1                   ;WRITE A UNIQUE NUMBER TO IT
8154 035424  010102                             4$:     MOV     R1,R2                   ;
8155 035426  072227  000010                             ASH     #10,R2                  ;
8156 035432  021427  000000                             CMP     (R4),#0                 ;
8157 035436  001401                                     BEQ     5$                      ;
8158 035440  104002                                     ERROR   +2              ;MMU ERROR
8159                                                                            ;
8160 035442  010224                             5$:     MOV     R2,(R4)+                ;
8161 035444  005201                                     INC     R1                      ;
8162 035446  020427  172400                             CMP     R4,#172400              ;
8163 035452  001364                                     BNE     4$                      ;
8164 035454  012704  177600                             MOV     #177600,R4              ;
```

```
035460  010102                  6$:     MOV     R1,R2                   ;
035462  072227  000010                  ASH     #10,R2                  ;
035466  021427  000000                  CMP     (R4),#0                 ;
035472  001401                          BEQ     7$                      ;
035474  104002                          ERROR   +2              ;MMU ERROR
                                                                ;
035476  010224                  7$:     MOV     R2,(R4)+                ;
035500  005201                          INC     R1                      ;
035502  020427  177700                  CMP     R4,#177700              ;
035506  001364                          BNE     6$                      ;
035510  012704  003062                  MOV     #FLO,R4                 ;CHECK AC5 FOR ALL ZEROES THEN LOAD A 6
035514  012703  003052                  MOV     #FLOAT,R3               ;
035520  174014                          STD     AC0,(R4)                ;
035522  172405                          LDD     AC5,AC0                 ;
035524  174013                          STD     AC0,(R3)                ;
035526  012702  000004                  MOV     #4,R2                   ;
035532  022327  000000          8$:     CMP     (R3)+,#0                ;
035536  001401                          BEQ     9$                      ;
035540  104002                          ERROR   +2              ;MMU ERROR
                                                                ;
035542  005302                  9$:     DEC     R2                      ;
035544  001372                          BNE     8$                      ;
035546  012703  003052                  MOV     #FLOAT,R3               ;
035552  012713  000006                  MOV     #6,(R3)                 ;
035556  172413                          LDD     (R3),AC0                ;
035560  174005                          STD     AC0,AC5                 ;
035562  172404                          LDD     AC4,AC0                 ;CHECK AC4 FOR ALL ZEROES THEN LOAD A 5
035564  174013                          STD     AC0,(R3)                ;
035566  012702  000004                  MOV     #4,R2                   ;
035572  022327  000000          10$:    CMP     (R3)+,#0                ;
035576  001401                          BEQ     11$                     ;
035600  104002                          ERROR   +2              ;MMU ERROR
                                                                ;
035602  005302                  11$:    DEC     R2                      ;
035604  001372                          BNE     10$                     ;
035606  012703  003052                  MOV     #FLOAT,R3               ;
035612  012713  000005                  MOV     #5,(R3)                 ;
035616  172413                          LDD     (R3),AC0                ;
035620  174004                          STD     AC0,AC4                 ;
035622  012702  000004                  MOV     #4,R2                   ;CHECK AC0 FOR ALL ZEROES THEN LOAD A 1
035626  022427  000000          12$:    CMP     (R4)+,#0                ;
035632  001401                          BEQ     13$                     ;
035634  104002                          ERROR   +2              ;MMU ERROR
                                                                ;
035636  005302                  13$:    DEC     R2                      ;
035640  001372                          BNE     12$                     ;
035642  012713  000001                  MOV     #1,(R3)                 ;
035646  172413                          LDD     (R3),AC0                ;
035650  012704  003062                  MOV     #FLO,R4                 ;CHECK AC1 FOR ALL ZEROES THEN LOAD A 2
035654  012702  000004                  MOV     #4,R2                   ;
035660  174114                          STD     AC1,(R4)                ;
035662  022427  000000          14$:    CMP     (R4)+,#0                ;
035666  001401                          BEQ     15$                     ;
035670  104002                          ERROR   +2              ;MMU ERROR
                                                                ;
035672  005302                  15$:    DEC     R2                      ;
035674  001372                          BNE     14$                     ;
```

MEMORY MANAGEMENT TESTS

```
8222 035676  012713  000002                MOV     #2,(R3)                 ;
8223 035702  172513                        LDD     (R3),AC1                ;
8224 035704  012704  003062                MOV     #FLO,R4                 ;CHECK AC2 FOR ALL ZEROES THEN LOAD A 3
8225 035710  012702  000004                MOV     #4,R2                   ;
8226 035714  174214                        STD     AC2,(R4)                ;
8227 035716  022427  000000    16$:        CMP     (R4)+,#0                ;
8228 035722  001401                        BEQ     17$                     ;
8229 035724  104002                        ERROR   .2              ;MMU ERROR
8230                                                               ;
8231 035726  005302            17$:        DEC     R2                      ;
8232 035730  001372                        BNE     16$
8233 035732  012713  000003                MOV     #3,(R3)                 ;
8234 035736  172613                        LDD     (R3),AC2                ;
8235 035740  012704  003062                MOV     #FLO,R4                 ;CHECK AC3 FOR ALL ZEROES THEN LOAD A 4
8236 035744  012702  000004                MOV     #4,R2                   ;
8237 035750  174314                        STD     AC3,(R4)                ;
8238 035752  022427  000000    18$:        CMP     (R4)+,#0                ;
8239 035756  001401                        BEQ     19$                     ;
8240 035760  104002                        ERROR   .2              ;MMU ERROR
8241                                                               ;
8242 035762  005302            19$:        DEC     R2                      ;
8243 035764  001372                        BNE     18$                     ;
8244 035766  012713  000004                MOV     #4,(R3)                 ;
8245 035772  172713                        LDD     (R3),AC3                ;
8246 035774  012704  003062                MOV     #FLO,R4                 ;CHECK FPS FOR 100204 THEN LOAD IT WITH 200
8247 036000  170214                        STFPS   (R4)                    ;
8248 036002  022714  100204                CMP     #100204,(R4)            ;
8249 036006  001401                        BEQ     20$                     ;
8250 036010  104002                        ERROR   .2              ;MMU ERROR
8251                                                               ;
8252 036012  170127  000200    20$:        LDFPS   #200                    ;
8253 036016  012704  172200                MOV     #172200,R4              ;CHECK PDR, PAR FOR UNIQUE NUMBERS
8254 036022  012701  000001                MOV     #1,R1                   ;
8255 036026  010102            23$:        MOV     R1,R2                   ;
8256 036030  072227  000010                ASH     #10,R2                  ;
8257 036034  022402                        CMP     (R4)+,R2                ;
8258 036036  001401                        BEQ     24$                     ;
8259 036040  104002                        ERROR   .2              ;MMU ERROR
8260                                                               ;
8261 036042  005201            24$:        INC     R1                      ;
8262 036044  020427  172400                CMP     R4,#172400              ;
8263 036050  001366                        BNE     23$                     ;
8264 036052  012704  177600                MOV     #177600,R4              ;
8265 036056  010102            25$:        MOV     R1,R2                   ;
8266 036060  072227  000010                ASH     #10,R2                  ;
8267 036064  022402                        CMP     (R4)+,R2                ;
8268 036066  001401                        BEQ     26$                     ;
8269 036070  104002                        ERROR   .2              ;MMU ERROR
8270                                                               ;
8271 036072  005201            26$:        INC     R1                      ;
8272 036074  020427  177700                CMP     R4,#177700              ;
8273 036100  001366                        BNE     25$                     ;
8274 036102  012701  003052                MOV     #FLOAT,R1               ;CHECK AC5 FOR #6
8275 036106  012704  003062                MOV     #FLO,R4                 ;
8276 036112  174014                        STD     AC0,(R4)                ;
8277 036114  172405                        LDD     AC5,AC0                 ;
8278 036116  174011                        STD     AC0,(R1)                ;
```

MEMORY MANAGEMENT TESTS

```
8279 036120  022127  000006              CMP     (R1)+,#6                ;
8280 036124  001401                      BEQ     27$                     ;
8281 036126  104002                      ERROR   +2                  ;MMU ERROR
8282                                                                 ;
8283 036130  012703  000003      27$:    MOV     #3,R3                   ;
8284 036134  022127  000000      28$:    CMP     (R1)+,#0                ;
8285 036140  001401                      BEQ     29$                     ;
8286 036142  104002                      ERROR   +2                  ;MMU ERROR
8287                                                                 ;
8288 036144  005303              29$:    DEC     R3                      ;
8289 036146  001372                      BNE     28$                     ;
8290 036150  012701  003052              MOV     #FLOAT,R1               ;CHECK AC4 FOR #5
8291 036154  172404                      LDD     AC4,AC0                 ;
8292 036156  174011                      STD     AC0,(R1)                ;
8293 036160  022127  000005              CMP     (R1)+,#5                ;
8294 036164  001401                      BEQ     30$                     ;
8295 036166  104002                      ERROR   +2                  ;MMU ERROR
8296                                                                 ;
8297 036170  012703  000003      30$:    MOV     #3,R3                   ;
8298 036174  022127  000000      31$:    CMP     (R1)+,#0                ;
8299 036200  001401                      BEQ     32$                     ;
8300 036202  104002                      ERROR   +2                  ;MMU ERROR
8301                                                                 ;
8302 036204  005303              32$:    DEC     R3                      ;
8303 036206  001372                      BNE     31$                     ;
8304 036210  022427  000001              CMP     (R4)+,#1                ;CHECK AC0 FOR #1
8305 036214  001401                      BEQ     33$                     ;
8306 036216  104002                      ERROR   +2                  ;MMU ERROR
8307                                                                 ;
8308 036220  012703  000003      33$:    MOV     #3,R3                   ;
8309 036224  022427  000000      34$:    CMP     (R4)+,#0                ;
8310 036230  001401                      BEQ     35$                     ;
8311 036232  104002                      ERROR   +2                  ;MMU ERROR
8312                                                                 ;
8313 036234  005303              35$:    DEC     R3                      ;
8314 036236  001372                      BNE     34$                     ;
8315 036240  012701  003052              MOV     #FLOAT,R1               ;CHECK AC1 FOR #2
8316 036244  174111                      STD     AC1,(R1)                ;
8317 036246  022127  000002              CMP     (R1)+,#2                ;
8318 036252  001401                      BEQ     36$                     ;
8319 036254  104002                      ERROR   +2                  ;MMU ERROR
8320                                                                 ;
8321 036256  012703  000003      36$:    MOV     #3,R3                   ;
8322 036262  022127  000000      37$:    CMP     (R1)+,#0                ;
8323 036266  001401                      BEQ     38$                     ;
8324 036270  104002                      ERROR   +2                  ;MMU ERROR
8325                                                                 ;
8326 036272  005303              38$:    DEC     R3                      ;
8327 036274  001372                      BNE     37$                     ;
8328 036276  012701  003052              MOV     #FLOAT,R1               ;CHECK AC2 FOR #3
8329 036302  174211                      STD     AC2,(R1)                ;
8330 036304  022127  000003              CMP     (R1)+,#3                ;
8331 036310  001401                      BEQ     39$                     ;
8332 036312  104002                      ERROR   +2                  ;MMU ERROR
8333                                                                 ;
8334 036314  012703  000003      39$:    MOV     #3,R3                   ;
8335 036320  022127  000000      40$:    CMP     (R1)+,#0                ;
```

MEMORY MANAGEMENT TESTS

```
 8336 036324  001401                        BEQ     41$                     ;
 8337 036326  104002                        ERROR   .2                      ;MMU ERROR
 8338                                                                       ;
 8339 036330  005303                 41$:   DEC     R3                      ;
 8340 036332  001372                        BNE     40$                     ;
 8341 036334  012701  003052                MOV     @FLOAT,R1               ;CHECK AC3 FOR @4
 8342 036340  174311                        STD     AC3,(R1)                ;
 8343 036342  022127  000004                CMP     (R1)+,@4                ;
 8344 036346  001401                        BEQ     42$                     ;
 8345 036350  104002                        ERROR   .2                      ;MMU ERROR
 8346                                                                       ;
 8347 036352  012703  000003         42$:   MOV     @3,R3                   ;
 8348 036356  022127  000000         43$:   CMP     (R1)+,@0                ;
 8349 036362  001401                        BEQ     44$                     ;
 8350 036364  104002                        ERROR   .2                      ;MMU ERROR
 8351                                                                       ;
 8352 036366  005303                 44$:   DEC     R3                      ;
 8353 036370  001372                        BNE     43$                     ;
 8354 036372  020527  000000                CMP     R5,@0                   ;IS TIME OUT FLAG 0
 8355 036376  001401                        BEQ     45$                     ;YES GO ON
 8356 036400  104002                        ERROR   .2                      ;MMU ERROR
 8357                                                                       ;NO GO TO ERROR
 8358 036402  012637  000004         45$:   MOV     (SP)+,@@4               ;RESTORE TIME OUT VECTOR
 8359 036406  012637  000246                MOV     (SP)+,@@246             ;RESTORE FP VECTOR
 8360 036412  012637  000244                MOV     (SP)+,@@244             ;
 8361
 8364 036416                         TSMMU3:
 8365                                ;      WRITE ALL PARS/PDRS WITH ONES THEN ZEROS
 8366 036416  005037  177572                CLR     @@177572                ;MMU OFF
 8367 036422  005037  003030                CLR     @@FLAG                  ;CLEAR MMU ABORT FLAG
 8368 036426  012703  172200                MOV     @172200,R3              ;LOAD ALL PARS AND PDRS WITH ONES
 8369 036432  012723  177777         1$:    MOV     @177777,(R3)+           ;
 8370 036436  020327  172400                CMP     R3,@172400              ;
 8371 036442  001373                        BNE     1$                      ;
 8372 036444  012703  177600                MOV     @177600,R3              ;
 8373 036450  012723  177777         2$:    MOV     @177777,(R3)+           ;
 8374 036454  020327  177700                CMP     R3,@177700              ;
 8375 036460  001373                        BNE     2$                      ;
 8376 036462  012703  172200                MOV     @172200,R3              ;CHECK SPDRS FOR ONES
 8377 036466  022327  177416         3$:    CMP     (R3)+,@177416           ;
 8378 036472  001401                        BEQ     4$                      ;
 8379 036474  104002                        ERROR   .2                      ;MMU ERROR
 8380                                                                       ;
 8381 036476  020327  172240         4$:    CMP     R3,@172240              ;
 8382 036502  001371                        BNE     3$                      ;
 8383 036504  022327  177777         5$:    CMP     (R3)+,@177777           ;CHECK SPARS FOR ONES
 8384 036510  001401                        BEQ     6$                      ;
 8385 036512  104002                        ERROR   .2                      ;MMU ERROR
 8386                                                                       ;
 8387 036514  020327  172300         6$:    CMP     R3,@172300              ;
 8388 036520  001371                        BNE     5$                      ;
 8389 036522  022327  177416         7$:    CMP     (R3)+,@177416           ;CHECK KPDRS FOR ONES
 8390 036526  001401                        BEQ     8$                      ;
 8391 036530  104002                        ERROR   .2                      ;MMU ERROR
 8392                                                                       ;
 8393 036532  020327  172340         8$:    CMP     R3,@172340              ;
 8394 036536  001371                        BNE     7$                      ;
```

MEMORY MANAGEMENT TESTS

```
8395 036540  022327  177777          9$:     CMP     (R3)+,#177777           ;CHECK KPARS FOR ONES
8396 036544  001401                          BEQ     10$                     ;
8397 036546  104002                          ERROR   .+2             ;MMU ERROR
8398                                                                 ;
8399 036550  020327  172400          10$:    CMP     R3,#172400              ;
8400 036554  001371                          BNE     9$                      ;
8401 036556  012703  177600                  MOV     #177600,R3              ;CHECK UPDRS FOR ONES
8402 036562  022327  177416          11$:    CMP     (R3)+,#177416           ;
8403 036566  001401                          BEQ     12$                     ;
8404 036570  104002                          ERROR   .+2             ;MMU ERROR
8405                                                                 ;
8406 036572  020327  177640          12$:    CMP     R3,#177640              ;
8407 036576  001371                          BNE     11$                     ;
8408 036600  022327  177777          13$:    CMP     (R3)+,#177777           ;CHECK UPARS FOR ONES
8409 036604  001401                          BEQ     14$                     ;
8410 036606  104002                          ERROR   .+2             ;MMU ERROR
8411                                                                 ;
8412 036610  020327  177700          14$:    CMP     R3,#177700              ;
8413 036614  001371                          BNE     13$                     ;
8414 036616  012703  172200                  MOV     #172200,R3              ;LOAD ALL PARS AND PDRS WITH ZEROES
8415 036622  012723  000000          15$:    MOV     #0,(R3)+                ;
8416 036626  020327  172400                  CMP     R3,#172400              ;
8417 036632  001373                          BNE     15$                     ;
8418 036634  012703  177600                  MOV     #177600,R3              ;
8419 036640  012723  000000          16$:    MOV     #0,(R3)+                ;
8420 036644  020327  177700                  CMP     R3,#177700              ;
8421 036650  001373                          BNE     16$                     ;
8422 036652  012703  172200                  MOV     #172200,R3              ;CHECK ALL PARS AND PDRS FOR ZEROES
8423 036656  022327  000000          17$:    CMP     (R3)+,#0                ;
8424 036662  001401                          BEQ     18$                     ;
8425 036664  104002                          ERROR   .+2             ;MMU ERROR
8426                                                                 ;
8427 036666  020327  172400          18$:    CMP     R3,#172400              ;
8428 036672  001371                          BNE     17$                     ;
8429 036674  012703  177600                  MOV     #177600,R3              ;
8430 036700  022327  000000          19$:    CMP     (R3)+,#0                ;
8431 036704  001401                          BEQ     20$                     ;
8432 036706  104002                          ERROR   .+2             ;MMU ERROR
8433                                                                 ;
8434 036710  020327  177700          20$:    CMP     R3,#177700              ;
8435 036714  001371                          BNE     19$                     ;
8436
8439 036716                          TSMMU4:
8440                                 ;       TEST FOR ADJACENT SHORTS IN PARS/PDRS
8441 036716  005037  177572                  CLR     @#177572                ;MMU OFF
8442 036722  005067  144102                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
8443 036726  012700  172200                  MOV     #172200,R0              ;LOAD SPDRS WITH ALTERNATING PATTERN
8444 036732  012720  052404          1$:     MOV     #52404,(R0)+            ;
8445 036736  012720  125012                  MOV     #125012,(R0)+           ;
8446 036742  020027  172240                  CMP     R0,#172240              ;
8447 036746  001371                          BNE     1$                      ;
8448 036750  012720  125252          2$:     MOV     #125252,(R0)+           ;LOAD SPARS WITH ALTERNATING PATTERN
8449 036754  012720  052525                  MOV     #52525,(R0)+            ;
8450 036760  020027  172300                  CMP     R0,#172300              ;
8451 036764  001371                          BNE     2$                      ;
8452 036766  012720  052404          3$:     MOV     #52404,(R0)+            ;LOAD KPDRS WITH ALTERNATING PATTERN
8453 036772  012720  125012                  MOV     #125012,(R0)+           ;
```

MEMORY MANAGEMENT TESTS

```
8454 036776  020027  172340            CMP     R0,#172340              ;
8455 037002  001371                    BNE     3$                      ;
8456 037004  012720  125252      4$:   MOV     #125252,(R0)+           ;LOAD KPARS WITH ALTERNATING PATTERN
8457 037010  012720  052525            MOV     #52525,(R0)+            ;
8458 037014  020027  172400            CMP     R0,#172400              ;
8459 037020  001371                    BNE     4$                      ;
8460 037022  012700  177600            MOV     #177600,R0              ;LOAD UPDRS WITH ALTERNATING PATTERN
8461 037026  012720  052404      5$:   MOV     #52404,(R0)+            ;
8462 037032  012720  125012            MOV     #125012,(R0)+           ;
8463 037036  020027  177640            CMP     R0,#177640              ;
8464 037042  001371                    BNE     5$                      ;
8465 037044  012720  125252      6$:   MOV     #125252,(R0)+           ;LOAD UPARS WITH ALTERNATING PATTERN
8466 037050  012720  052525            MOV     #52525,(R0)+            ;
8467 037054  020027  177700            CMP     R0,#177700              ;
8468 037060  001371                    BNE     6$                      ;
8469                              ;
8470 037062  012703  172200            MOV     #172200,R3              ;CHECK SPDRS
8471 037066  022327  052404      7$:   CMP     (R3)+,#52404            ;
8472 037072  001401                    BEQ     8$                      ;
8473 037074  104002                    ERROR   +2              ;MMU ERROR
8474                                    ;
8475 037076  022327  125012      8$:   CMP     (R3)+,#125012           ;
8476 037102  001401                    BEQ     9$                      ;
8477 037104  104002                    ERROR   +2              ;MMU ERROR
8478                                    ;
8479 037106  020327  172240      9$:   CMP     R3,#172240              ;
8480 037112  001365                    BNE     7$                      ;
8481 037114  022327  125252      10$:  CMP     (R3)+,#125252           ;CHECK SPARS
8482 037120  001401                    BEQ     11$                     ;
8483 037122  104002                    ERROR   +2              ;MMU ERROR
8484                                    ;
8485 037124  022327  052525      11$:  CMP     (R3)+,#52525            ;
8486 037130  001401                    BEQ     12$                     ;
8487 037132  104002                    ERROR   +2              ;MMU ERROR
8488                                    ;
8489 037134  020327  172300      12$:  CMP     R3,#172300              ;
8490 037140  001365                    BNE     10$                     ;
8491 037142  022327  052404      13$:  CMP     (R3)+,#52404            ;CHECK KPDRS
8492 037146  001401                    BEQ     14$                     ;
8493 037150  104002                    ERROR   +2              ;MMU ERROR
8494                                    ;
8495 037152  022327  125012      14$:  CMP     (R3)+,#125012           ;
8496 037156  001401                    BEQ     15$                     ;
8497 037160  104002                    ERROR   +2              ;MMU ERROR
8498                                    ;
8499 037162  020327  172340      15$:  CMP     R3,#172340              ;
8500 037166  001365                    BNE     13$                     ;
8501 037170  022327  125252      16$:  CMP     (R3)+,#125252           ;CHECK KPARS
8502 037174  001401                    BEQ     17$                     ;
8503 037176  104002                    ERROR   +2              ;MMU ERROR
8504                                    ;
8505 037200  022327  052525      17$:  CMP     (R3)+,#52525            ;
8506 037204  001401                    BEQ     18$                     ;
8507 037206  104002                    ERROR   +2              ;MMU ERROR
8508                                    ;
8509 037210  020327  172400      18$:  CMP     R3,#172400              ;
8510 037214  001365                    BNE     16$                     ;
```

L12

MEMORY MANAGEMENT TESTS

```
8511 037216  012703  177600              MOV     #177600,R3              ;CHECK UPDRS
8512 037222  022327  052404      19$:    CMP     (R3)+,#52404            ;
8513 037226  001401                      BEQ     20$                     ;
8514 037230  104002                      ERROR   +2              ;MMU ERROR
8515                                                             ;
8516 037232  022327  125012      20$:    CMP     (R3)+,#125012           ;
8517 037236  001401                      BEQ     21$                     ;
8518 037240  104002                      ERROR   +2              ;MMU ERROR
8519                                                             ;
8520 037242  020327  177640      21$:    CMP     R3,#177640              ;
8521 037246  001365                      BNE     19$                     ;
8522 037250  022327  125252      22$:    CMP     (R3)+,#125252           ;CHECK UPARS
8523 037254  001401                      BEQ     23$                     ;
8524 037256  104002                      ERROR   +2              ;MMU ERROR
8525                                                             ;
8526 037260  022327  052525      23$:    CMP     (R3)+,#52525            ;
8527 037264  001401                      BEQ     24$                     ;
8528 037266  104002                      ERROR   +2              ;MMU ERROR
8529                                                             ;
8530 037270  020327  177700      24$:    CMP     R3,#177700              ;
8531 037274  001365                      BNE     22$                     ;
8532                             ;
8533                             ;REVERSE ALTERNATING PATTERN
8534                             ;
8535 037276  012700  172200              MOV     #172200,R0              ;LOAD SPDRS WITH REVERSE PATTERN
8536 037302  012720  125012      25$:    MOV     #125012,(R0)+           ;
8537 037306  012720  052404              MOV     #52404,(R0)+            ;
8538 037312  020027  172240              CMP     R0,#172240              ;
8539 037316  001371                      BNE     25$                     ;
8540 037320  012720  052525      26$:    MOV     #52525,(R0)+            ;LOAD SPARS WITH REVERSE PATTERN
8541 037324  012720  125252              MOV     #125252,(R0)+           ;
8542 037330  020027  172300              CMP     R0,#172300              ;
8543 037334  001371                      BNE     26$                     ;
8544 037336  012720  125012      27$:    MOV     #125012,(R0)+           ;LOAD KPDRS WITH REVERSE PATTERN
8545 037342  012720  052404              MOV     #52404,(R0)+            ;
8546 037346  020027  172340              CMP     R0,#172340              ;
8547 037352  001371                      BNE     27$                     ;
8548 037354  012720  052525      28$:    MOV     #52525,(R0)+            ;LOAD KPARS WITH REVERSE PATTERN
8549 037360  012720  125252              MOV     #125252,(R0)+           ;
8550 037364  020027  172400              CMP     R0,#172400              ;
8551 037370  001371                      BNE     28$                     ;
8552 037372  012700  177600              MOV     #177600,R0              ;LOAD UPDRS WITH REVERSE PATTERN
8553 037376  012720  125012      29$:    MOV     #125012,(R0)+           ;
8554 037402  012720  052404              MOV     #52404,(R0)+            ;
8555 037406  020027  177640              CMP     R0,#177640              ;
8556 037412  001371                      BNE     29$                     ;
8557 037414  012720  052525      30$:    MOV     #52525,(R0)+            ;LOAD UPARS WITH REVERSE PATTERN
8558 037420  012720  125252              MOV     #125252,(R0)+           ;
8559 037424  020027  177700              CMP     R0,#177700              ;
8560 037430  001371                      BNE     30$                     ;
8561                             ;
8562 037432  012703  172200              MOV     #172200,R3              ;CHECK SPDRS
8563 037436  022327  125012      31$:    CMP     (R3)+,#125012           ;
8564 037442  001401                      BEQ     32$                     ;
8565 037444  104002                      ERROR   +2              ;MMU ERROR
8566                                                             ;
8567 037446  022327  052404      32$:    CMP     (R3)+,#52404            ;
```

MEMORY MANAGEMENT TESTS

```
8568 037452  001401                    BEQ     33$                     ;
8569 037454  104002                    ERROR   +2                      ;MMU ERROR
8570                                                                   ;
8571 037456  020327  172240      33$:   CMP     R3,#172240              ;
8572 037462  001365                    BNE     31$                     ;
8573 037464  022327  052525      34$:   CMP     (R3)+,#52525            ;CHECK SPARS
8574 037470  001401                    BEQ     35$                     ;
8575 037472  104002                    ERROR   +2                      ;MMU ERROR
8576                                                                   ;
8577 037474  022327  125252      35$:   CMP     (R3)+,#125252           ;
8578 037500  001401                    BEQ     36$                     ;
8579 037502  104002                    ERROR   +2                      ;MMU ERROR
8580                                                                   ;
8581 037504  020327  172300      36$:   CMP     R3,#172300              ;
8582 037510  001365                    BNE     34$                     ;
8583 037512  022327  125012      37$:   CMP     (R3)+,#125012           ;CHECK KPDRS
8584 037516  001401                    BEQ     38$                     ;
8585 037520  104002                    ERROR   +2                      ;MMU ERROR
8586                                                                   ;
8587 037522  022327  052404      38$:   CMP     (R3)+,#52404            ;
8588 037526  001401                    BEQ     39$                     ;
8589 037530  104002                    ERROR   +2                      ;MMU ERROR
8590                                                                   ;
8591 037532  020327  172340      39$:   CMP     R3,#172340              ;
8592 037536  001365                    BNE     37$                     ;
8593 037540  022327  052525      40$:   CMP     (R3)+,#52525            ;CHECK KPARS
8594 037544  001401                    BEQ     41$                     ;
8595 037546  104002                    ERROR   +2                      ;MMU ERROR
8596                                                                   ;
8597 037550  022327   25252      41$:   CMP     (R3)+,#125252           ;
8598 037554  001401                    BEQ     42$                     ;
8599 037556  104002                    ERROR   +2                      ;MMU ERROR
8600                                                                   ;
8601 037560  020327  172400      42$:   CMP     R3,#172400              ;
8602 037564  001365                    BNE     40$                     ;
8603 037566  012703  177600            MOV     #177600,R3              ;CHECK UPDRS
8604 037572  022327  125012      43$:   CMP     (R3)+,#125012           ;
8605 037576  001401                    BEQ     44$                     ;
8606 037600  104002                    ERROR   +2                      ;MMU ERROR
8607                                                                   ;
8608 037602  022327  052404      44$:   CMP     (R3)+,#52404            ;
8609 037606  001401                    BEQ     45$                     ;
8610 037610  104002                    ERROR   +2                      ;MMU ERROR
8611                                                                   ;
8612 037612  020327  177640      45$:   CMP     R3,#177640              ;
8613 037616  001365                    BNE     43$                     ;
8614 037620  022327  052525      46$:   CMP     (R3)+,#52525            ;CHECK UPARS
8615 037624  001401                    BEQ     47$                     ;
8616 037626  104002                    ERROR   +2                      ;MMU ERROR
8617                                                                   ;
8618 037630  022327  125252      47$:   CMP     (R3)+,#125252           ;
8619 037634  001401                    BEQ     48$                     ;
8620 037636  104002                    ERROR   +2                      ;MMU ERROR
8621                                                                   ;
8622 037640  020327  177700      48$:   CMP     R3,#177700              ;
8623 037644  001365                    BNE     46$                     ;
8624
```

MEMORY MANAGEMENT TESTS

```
                                        TSMMU5:
                                        ;       TEST MMR0 ABORT BITS
037646                                  ;
037646  012737  160000  177572          ;       MOV     #160000,@#177572        ;LOAD MMR0<15:13>=111
037654  005067  143150                  ;       CLR     FLAG                    ;CLEAR MMU ABORT FLAG
037660  013700  177572                  ;       MOV     @#SR0,R0                ;SAVE SR0 IN R0
037664  042700  000176                  ;       BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0
037670  020027  160000                  ;       CMP     R0,#160000              ;CHECK MMR0
037674  001401                          ;       BEQ     1$                      ;
037676  104002                          ;       ERROR   +2              ;MMU ERROR
                                        ;
037700  005037  177572          1$:     ;       CLR     @#177572                ;LOAD MMR0=0
037704  013700  177572                  ;       MOV     @#SR0,R0                ;SAVE SR0 IN R0
037710  042700  000176                  ;       BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0
037714  020027  000000                  ;       CMP     R0,#0                   ;CHECK MMR0
037720  001401                          ;       BEQ     2$                      ;
037722  104002                          ;       ERROR   +2              ;MMU ERROR
                                        ;
037724  012737  120000  177572  2$:     ;       MOV     #120000,@#177572        ;LOAD MMR0<15:13>=101
037732  013700  177572                  ;       MOV     @#SR0,R0                ;SAVE SR0 IN R0
037736  042700  000176                  ;       BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0.
037742  020027  120000                  ;       CMP     R0,#120000              ;CHECK MMR0
037746  001401                          ;       BEQ     3$                      ;
037750  104002                          ;       ERROR   +2              ;MMU ERROR
                                        ;
037752  012737  040000  177572  3$:     ;       MOV     #40000,@#177572         ;LOAD MMR0<15:13>=010
037760  013700  177572                  ;       MOV     @#SR0,R0                ;SAVE SR0 IN R0
037764  042700  000176                  ;       BIC     #176,R0                 ;CLEAR UNDEFINED BITS FROM SR0.
037770  020027  040000                  ;       CMP     R0,#40000               ;CHECK MMR0
037774  001401                          ;       BEQ     4$                      ;
037776  104002                          ;       ERROR   +2              ;MMU ERROR
040000                                  4$:
040000                                  TSMMU6:
                                        ;       TEST MMR3 BITS 5-0
040000  005037  177572                  ;       CLR     @#177572                ;MMU OFF
040004  005067  143020                  ;       CLR     FLAG                    ;CLEAR MMU ABORT FLAG
040010  012737  000077  172516          ;       MOV     #77,@#172516            ;LOAD MMR3<5:0>=77
040016  023727  172516  000077          ;       CMP     @#172516,#77            ;CHECK MMR3
040024  001401                          ;       BEQ     1$                      ;
040026  104002                          ;       ERROR   +2              ;MMU ERROR
040030  005037  172516          1$:     ;       CLR     @#172516                ;LOAD MMR3<5:0>=0
040034  023727  172516  000000          ;       CMP     @#172516,#0             ;CHECK MMR3
040042  001401                          ;       BEQ     2$                      ;
040044  104002                          ;       ERROR   +2              ;MMU ERROR
040046  012737  000052  172516  2$:     ;       MOV     #52,@#172516            ;LOAD MMR3<5:0>=52
040054  023727  172516  000052          ;       CMP     @#172516,#52            ;CHECK MMR3
040062  001401                          ;       BEQ     3$                      ;
040064  104002                          ;       ERROR   +2              ;MMU ERROR
040066  012737  000025  172516  3$:     ;       MOV     #25,@#172516            ;LOAD MMR3<5:0>=25
040074  023727  172516  000025          ;       CMP     @#172516,#25            ;CHECK MMR3
040102  001401                          ;       BEQ     4$                      ;
040104  104002                          ;       ERROR   +2              ;MMU ERROR
040106                                  4$:
040106                                  TSMM6A:
                                        ;       TEST MFPI (MOVE FROM PREVIOUS INST SPACE)
040106  005037  177572                  ;       CLR     @#177572                ;MMU OFF
040112  005037  003030                  ;       CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
040116  012737  140000  177776          ;       MOV     #140000,@#177776        ;POINT TO USER SPACE
```

MEMORY MANAGEMENT TESTS

```
 8688 040124  012706  001000                    MOV     #STBOT,SP               ;INIT THE USER STACK POINTER
 8689 040130  010637  003040                    MOV     R6,@#SAVUSE             ;SAVE USER SP
 8690 040134  012737  040000  177776            MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 8691 040142  012706  001000                    MOV     #STBOT,SP               ;INIT THE SUPERVISOR STACK POINTER
 8692 040146  010637  003036                    MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
 8693 040152  012737  030000  177776            MOV     #30000,@#177776         ;SETUP PSW
 8694 040160  004767  074702                    JSR     PC,MMU                  ;INIT MMU
 8695 040164  012737  000027  172516            MOV     #27,@#172516            ;SETUP MMR3
 8696 040172  013746  000244                    MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
 8697 040176  012746  177777                    MOV     #177777,-(SP)           ;PUT KNOWN DATA ON TOP OF STACK
 8698 040202  012737  135072  000244            MOV     #135072,@#244           ;SETUP DATA AT TEST LOCATION
 8699 040210  012767  077400  137402            MOV     #77400,UDPDR0           ;SETUP UDPDR0 TO ABORT
 8700 040216  012703  000244                    MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
 8701 040222  005237  177572                    INC     @#177572                ;TURN MMU ON
 8702 040226  006523                            MFPI    (R3)+                   ; TEST INSTRUCTION
 8703 040230  022737  030010  177776            CMP     #30010,@#177776         ;IS PSW CORRECT
 8704 040236  001401                            BEQ     1$                      ;YES GO ON
 8705 040240  104002                            ERROR   .2              ;MMU ERROR
 8706                                           ;NO GO TO ERROR
 8707 040242  005037  177572          1$:       CLR     @#177572                ;TURN MMU OFF
 8708 040246  012737  140000  177776            MOV     #140000,@#177776        ;POINT TO USER SPACE
 8709 040254  020637  003040                    CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
 8710 040260  001401                            BEQ     100$                    ;YES GO ON
 8711 040262  104002                            ERROR   .2              ;MMU ERROR
 8712                                           ;NO GO TO ERROR
 8713 040264  012737  040000  177776  100$:     MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 8714 040272  020637  003036                    CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
 8715 040276  001401                            BEQ     200$                    ;YES GO ON
 8716 040300  104002                            ERROR   .2              ;MMU ERROR
 8717                                           ;NO GO TO ERROR
 8718 040302  023727  000244  135072  200$:     CMP     @#244,#135072           ;IS TEST DATA OK
 8719 040310  001401                            BEQ     2$                      ;YES GO ON
 8720 040312  104002                            ERROR   .2              ;MMU ERROR
 8721                                           ;NO GO TO ERROR
 8722 040314  020327  000246          2$:       CMP     R3,#246                 ;IS R3 CORRECT
 8723 040320  001401                            BEQ     3$                      ;YES GO ON
 8724 040322  104002                            ERROR   .2              ;MMU ERROR
 8725                                           ;NO GO TO ERROR
 8726 040324  005037  177776          3$:       CLR     @#177776                ;SET PSW TO KERNEL MODE
 8727 040330  022627  135072                    CMP     (SP)+,#135072           ;IS KERNEL STACK CORRECT
 8728 040334  001401                            BEQ     4$                      ;YES GO ON
 8729 040336  104002                            ERROR   .2              ;MMU ERROR
 8730                                           ;NO GO TO ERROR
 8731 040340  021627  177777          4$:       CMP     (SP),#177777            ;IS STACK CORRECT
 8732 040344  001401                            BEQ     5$                      ;YES GO ON
 8733 040346  104002                            ERROR   .2              ;MMU ERROR
 8734                                           ;NO GO TO ERROR
 8735 040350  012737  030017  177776  5$:       MOV     #30017,@#177776         ;SETUP PSW
 8736 040356  012737  173621  000244            MOV     #173621,@#244           ;SETUP TEST LOCATION
 8737 040364  012701  000244                    MOV     #244,R1                 ;SETUP R1
 8738 040370  005237  177572                    INC     @#177572                ;TURN MMU ON
 8739 040374  006511                            MFPI    (R1)                    ;TEST INSTRUCTION
 8740 040376  022737  030011  177776            CMP     #30011,@#177776         ;IS PSW CORRECT
 8741 040404  001401                            BEQ     300$                    ;YES GO ON
 8742 040406  104002                            ERROR   .2              ;MMU ERROR
 8743                                           ;NO GO TO ERROR
 8744 040410  005037  177572          300$:     CLR     @#177572                ;TURN MMU OFF
```

MEMORY MANAGEMENT TESTS

```
8745 040414  023727  000244  173621          CMP     @#244,#173621           ;IS TEST LOCATION CORRECT
8746 040422  001401                          BEQ     301$                    ;YES GO ON
8747 040424  104002                          ERROR   .2                  ;MMU ERROR
8748                                                                     ;NO GO TO ERROR
8749 040426  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8750 040432  001401                          BEQ     302$                    ;YES GO ON
8751 040434  104002                          ERROR   .2                  ;MMU ERROR
8752                                                                     ;NO GO TO ERROR
8753 040436  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8754 040442  022627  173621                  CMP     (SP)+,#173621           ;IS STACK CORRECT
8755 040446  001401                          BEQ     303$                    ;YES GO ON
8756 040450  104002                          ERROR   .2                  ;MMU ERROR
8757                                                                     ;NO GO TO ERROR
8758 040452  021627  177777          303$:   CMP     (SP),#177777            ;IS STACK CORRECT
8759 040456  001401                          BEQ     304$                    ;YES GO ON
8760 040460  104002                          ERROR   .2                  ;MMU ERROR
8761                                                                     ;NO GO TO ERROR
8762 040462  005003                  304$:   CLR     R3                      ;SETUP SOURCE FOR NEXT TEST
8763 040464  005237  177572                  INC     @#177572                ;TURN MMU ON
8764 040470  006503                          MFPI    R3                      ; TEST INSTRUCTION
8765 040472  022737  000004  177776          CMP     #4,@#177776             ;IS PSW CORRECT
8766 040500  001401                          BEQ     6$                      ;YES GO ON
8767 040502  104002                          ERROR   .2                  ;MMU ERROR
8768                                                                     ;NO GO TO ERROR
8769 040504  005037  177572          6$:     CLR     @#177572                ;TURN MMU OFF
8770 040510  020327  000000                  CMP     R3,#0                   ;IS R3 CORRECT
8771 040514  001401                          BEQ     7$                      ;YES GO ON
8772 040516  104002                          ERROR   .2                  ;MMU ERROR
8773                                                                     ;NO GO TO ERROR
8774 040520  022627  000000          7$:     CMP     (SP)+,#0                ;IS STACK CORRECT
8775 040524  001401                          BEQ     8$                      ;YES GO ON
8776 040526  104002                          ERROR   .2                  ;MMU ERROR
8777                                                                     ;NO GO TO ERROR
8778 040530  022627  177777          8$:     CMP     (SP)+,#177777           ;IS STACK CORRECT
8779 040534  001401                          BEQ     9$                      ;YES GO ON
8780 040536  104002                          ERROR   .2                  ;MMU ERROR
8781                                                                     ;NO GO TO ERROR
8782 040540  012637  000244          9$:     MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
8783
8784                                 ;
8787 040544                          TSMM68:
8788                                 ;       TEST MFPD (MOVE FROM PREVIOUS DATA SPACE)
8789 040544  005037  177572                  CLR     @#177572                ;MMU OFF
8790 040550  005037  003030                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
8791 040554  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8792 040562  010637  003040                  MOV     R6,@#SAVUSE             ;SAVE USER SP
8793 040566  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8794 040574  010637  003036                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
8795 040600  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
8796 040606  004767  074254                  JSR     PC,MMU                  ;INIT MMU
8797 040612  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
8798 040620  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
8799 040624  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON TOP OF STACK
8800 040630  012737  157002  000244          MOV     #157002,@#244           ;SETUP DATA AT TEST LOCATION
8801 040636  012767  077400  136734          MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
8802 040644  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
8803 040650  005237  177572                  INC     @#177572                ;TURN MMU ON
```

MEMORY MANAGEMENT TESTS

```
8804 040654  106523                          MFPD    (R3)+                   ; TEST INSTRUCTION
8805 040656  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
8806 040664  001401                          BEQ     1$                      ;YES GO ON
8807 040666  104002                          ERROR   .2              ;MMU ERROR
8808                                                                 ;NO GO TO ERROR
8809 040670  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
8810 040674  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8811 040702  020637  003040                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8812 040706  001401                          BEQ     100$                    ;YES GO ON
8813 040710  104002                          ERROR   .2              ;MMU ERROR
8814                                                                 ;NO GO TO ERROR
8815 040712  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8816 040720  020637  003036                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8817 040724  001401                          BEQ     200$                    ;YES GO ON
8818 040726  104002                          ERROR   .2              ;MMU ERROR
8819                                                                 ;NO GO TO ERROR
8820 040730  023727  000244  157002  200$:   CMP     @#244,#157002           ;IS TEST DATA OK
8821 040736  001401                          BEQ     2$                      ;YES GO ON
8822 040740  104002                          ERROR   .2              ;MMU ERROR
8823                                                                 ;NO GO TO ERROR
8824 040742  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
8825 040746  001401                          BEQ     3$                      ;YES GO ON
8826 040750  104002                          ERROR   .2              ;MMU ERROR
8827                                                                 ;NO GO TO ERROR
8828 040752  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
8829 040756  022627  157002                  CMP     (SP)+,#157002           ;IS KERNEL STACK CORRECT
8830 040762  001401                          BEQ     4$                      ;YES GO ON
8831 040764  104002                          ERROR   .2              ;MMU ERROR
8832                                                                 ;NO GO TO ERROR
8833 040766  021627  177777          4$:     CMP     (SP),#177777            ;IS STACK CORRECT
8834 040772  001401                          BEQ     5$                      ;YES GO ON
8835 040774  104002                          ERROR   .2              ;MMU ERROR
8836                                                                 ;NO GO TO ERROR
8837 040776  012737  030017  177776  5$:     MOV     #30017,@#177776         ;SETUP PSW
8838 041004  012737  103456  000244          MOV     #103456,@#244           ;SETUP TEST LOCATION
8839 041012  012701  000244                  MOV     #244,R1                 ;SETUP R1
8840 041016  005237  177572                  INC     @#177572                ;TURN MMU ON
8841 041022  106511                          MFPD    (R1)                    ;TEST INSTRUCTION
8842 041024  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8843 041032  001401                          BEQ     300$                    ;YES GO ON
8844 041034  104002                          ERROR   .2              ;MMU ERROR
8845                                                                 ;NO GO TO ERROR
8846 041036  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
8847 041042  023727  000244  103456          CMP     @#244,#103456           ;IS TEST LOCATION CORRECT
8848 041050  001401                          BEQ     301$                    ;YES GO ON
8849 041052  104002                          ERROR   .2              ;MMU ERROR
8850                                                                 ;NO GO TO ERROR
8851 041054  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8852 041060  001401                          BEQ     302$                    ;YES GO ON
8853 041062  104002                          ERROR   .2              ;MMU ERROR
8854                                                                 ;NO GO TO ERROR
8855 041064  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8856 041070  022627  103456                  CMP     (SP)+,#103456           ;IS STACK CORRECT
8857 041074  001401                          BEQ     303$                    ;YES GO ON
8858 041076  104002                          ERROR   .2              ;MMU ERROR
8859                                                                 ;NO GO TO ERROR
8860 041100  021627  177777          303$:   CMP     (SP),#177777            ;IS STACK CORRECT
```

MEMORY MANAGEMENT TESTS

```
 8861 041104  001401                        BEQ     304$                    ;YES GO ON
 8862 041106  104002                        ERROR   .2                     ;MMU ERROR
 8863                                                                       ;NO GO TO ERROR
 8864 041110  012737  030017  177776  304$: MOV     #30017,@#177776         ;SETUP PSW
 8865 041116  012737  113672  000244        MOV     #113672,@#244           ;SETUP TEST LOCATION
 8866 041124  012701  000246                MOV     #246,R1                 ;SETUP R1
 8867 041130  005237  177572                INC     @#177572                ;TURN MMU ON
 8868 041134  106541                        MFPD    -(R1)                   ;TEST INSTRUCTION
 8869 041136  022737  030011  177776        CMP     #30011,@#177776         ;IS PSW CORRECT
 8870 041144  001401                        BEQ     400$                    ;YES GO ON
 8871 041146  104002                        ERROR   .2                      ;MMU ERROR
 8872                                                                       ;NO GO TO ERROR
 8873 041150  005037  177572          400$: CLR     @#177572                ;TURN MMU OFF
 8874 041154  023727  000244  113672        CMP     @#244,#113672           ;IS TEST LOCATION CORRECT
 8875 041162  001401                        BEQ     401$                    ;YES GO ON
 8876 041164  104002                        ERROR   .2                      ;MMU ERROR
 8877                                                                       ;NO GO TO ERROR
 8878 041166  020127  000244          401$: CMP     R1,#244                 ;IS R1 CORRECT
 8879 041172  001401                        BEQ     402$                    ;YES GO ON
 8880 041174  104002                        ERROR   .2                      ;MMU ERROR
 8881                                                                       ;NO GO TO ERROR
 8882 041176  005037  177776          402$: CLR     @#177776                ;SET PSW TO KERNEL MODE
 8883 041202  022627  113672                CMP     (SP)+,#113672           ;IS STACK CORRECT
 8884 041206  001401                        BEQ     403$                    ;YES GO ON
 8885 041210  104002                        ERROR   .2                      ;MMU ERROR
 8886                                                                       ;NO GO TO ERROR
 8887 041212  021627  177777          403$: CMP     (SP),#177777            ;IS STACK CORRECT
 8888 041216  001401                        BEQ     404$                    ;YES GO ON
 8889 041220  104002                        ERROR   .2                      ;MMU ERROR
 8890                                                                       ;NO GO TO ERROR
 8891 041222  005003                  404$: CLR     R3                      ;SETUP SOURCE FOR NEXT TEST
 8892 041224  005237  177572                INC     @#177572                ;TURN MMU ON
 8893 041230  106503                        MFPD    R3                      ;  TEST INSTRUCTION
 8894 041232  022737  000004  177776        CMP     #4,@#177776             ;IS PSW CORRECT
 8895 041240  001401                        BEQ     6$                      ;YES GO ON
 8896 041242  104002                        ERROR   .2                      ;MMU ERROR
 8897                                                                       ;NO GO TO ERROR
 8898 041244  005037  177572            6$: CLR     @#177572                ;TURN MMU OFF
 8899 041250  020327  000000                CMP     R3,#0                   ;IS R3 CORRECT
 8900 041254  001401                        BEQ     7$                      ;YES GO ON
 8901 041256  104002                        ERROR   .2                      ;MMU ERROR
 8902                                                                       ;NO GO TO ERROR
 8903 041260  022627  000000            7$: CMP     (SP)+,#0                ;IS STACK CORRECT
 8904 041264  001401                        BEQ     8$                      ;YES GO ON
 8905 041266  104002                        ERROR   .2                      ;MMU ERROR
 8906                                                                       ;NO GO TO ERROR
 8907 041270  022627  177777            8$: CMP     (SP)+,#177777           ;IS STACK CORRECT
 8908 041274  001401                        BEQ     9$                      ;YES GO ON
 8909 041276  104002                        ERROR   .2                      ;MMU ERROR
 8910                                                                       ;NO GO TO ERROR
 8911 041300  012637  000244            9$: MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
 8912
 8913                                ;
 8916 041304                        TSMM6C:
 8917                                ;       TEST MTPI (MOVE TO PREVIOUS INSTRUCTION SPACE)
 8918 041304  005037  177572                CLR     @#177572                ;MMU OFF
 8919 041310  005037  003030                CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
```

MEMORY MANAGEMENT TESTS

```
 8920 041314  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
 8921 041322  010637  003040                  MOV     R6,@#SAVUSE             ;SAVE USER SP
 8922 041326  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 8923 041334  010637  003036                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
 8924 041340  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 8925 041346  004767  073514                  JSR     PC,MMU                  ;INIT MMU
 8926 041352  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
 8927 041360  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
 8928 041364  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON STACK
 8929 041370  012746  120413                  MOV     #120413,-(SP)           ;PUT TEST DATA ON STACK
 8930 041374  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
 8931 041402  012767  077400  136210          MOV     #77400,UDPDR0           ;SETUP UDPDR0 TO ABORT
 8932 041410  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
 8933 041414  005237  177572                  INC     @#177572                ;TURN MMU ON
 8934 041420  006623                          MTPI    (R3)+                   ; TEST INSTRUCTION
 8935 041422  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
 8936 041430  001401                          BEQ     1$                      ;YES GO ON
 8937 041432  104002                          ERROR   .+2             ;MMU ERROR
 8938                                                                 ;NO GO TO ERROR
 8939 041434  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
 8940 041440  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
 8941 041446  020637  003040                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
 8942 041452  001401                          BEQ     100$                    ;YES GO ON
 8943 041454  104002                          ERROR   .+2             ;MMU ERROR
 8944                                                                 ;NO GO TO ERROR
 8945 041456  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 8946 041464  020637  003036                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
 8947 041470  001401                          BEQ     200$                    ;YES GO ON
 8948 041472  104002                          ERROR   .+2             ;MMU ERROR
 8949                                                                 ;NO GO TO ERROR
 8950 041474  023727  000244  120413  200$:   CMP     @#244,#120413           ;IS TEST LOCATION CORRECT
 8951 041502  001401                          BEQ     2$                      ;YES GO ON
 8952 041504  104002                          ERROR   .+2             ;MMU ERROR
 8953                                                                 ;NO GO TO ERROR
 8954 041506  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
 8955 041512  001401                          BEQ     3$                      ;YES GO ON
 8956 041514  104002                          ERROR   .+2             ;MMU ERROR
 8957                                                                 ;NO GO TO ERROR
 8958 041516  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
 8959 041522  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
 8960 041526  001401                          BEQ     4$                      ;YES GO ON
 8961 041530  104002                          ERROR   .+2             ;MMU ERROR
 8962                                                                 ;NO GO TO ERROR
 8963 041532  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
 8964 041540  012746  145121                  MOV     #145121,-(SP)           ;SETUP TEST DATA
 8965 041544  012701  000244                  MOV     #244,R1                 ;SETUP R1
 8966 041550  005237  177572                  INC     @#177572                ;TURN MMU ON
 8967 041554  006611                          MTPI    (R1)                    ;TEST INSTRUCTION
 8968 041556  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
 8969 041564  001401                          BEQ     300$                    ;YES GO ON
 8970 041566  104002                          ERROR   .+2             ;MMU ERROR
 8971                                                                 ;NO GO TO ERROR
 8972 041570  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
 8973 041574  023727  000244  145121          CMP     @#244,#145121           ;IS TEST LCCATION CORRECT
 8974 041602  001401                          BEQ     301$                    ;YES GO ON
 8975 041604  104002                          ERROR   .+2             ;MMU ERROR
 8976                                                                 ;NO GO TO ERROR
```

MEMORY MANAGEMENT TESTS

```
 8977 041606  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
 8978 041612  001401                          BEQ     302$                    ;YES GO ON
 8979 041614  104002                          ERROR   .+2                     ;MMU ERROR
 8980                                                                         ;NO GO TO ERROR
 8981 041616  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
 8982 041622  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
 8983 041626  001401                          BEQ     304$                    ;YES GO ON
 8984 041630  104002                          ERROR   .+2                     ;MMU ERROR
 8985                                                                         ;NO GO TO ERROR
 8986 041632  012737  030017  177776  304$:   MOV     #30017,@#177776         ;SETUP PSW
 8987 041640  012746  122347                  MOV     #122347,-(SP)           ;SETUP TEST DATA
 8988 041644  012701  000246                  MOV     #246,R1                 ;SETUP R1
 8989 041650  005237  177572                  INC     @#177572                ;TURN MMU ON
 8990 041654  006641                          MTPI    -(R1)                   ;TEST INSTRUCTION
 8991 041656  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
 8992 041664  001401                          BEQ     400$                    ;YES GO ON
 8993 041666  104002                          ERROR   .+2                     ;MMU ERROR
 8994                                                                         ;NO GO TO ERROR
 8995 041670  005037  177572          400$:   CLR     @#177572                ;TURN MMU OFF
 8996 041674  023727  000244  122347          CMP     @#244,#122347           ;IS TEST LOCATION CORRECT
 8997 041702  001401                          BEQ     401$                    ;YES GO ON
 8998 041704  104002                          ERROR   .+2                     ;MMU ERROR
 8999                                                                         ;NO GO TO ERROR
 9000 041706  020127  000244          401$:   CMP     R1,#244                 ;IS R1 CORRECT
 9001 041712  001401                          BEQ     402$                    ;YES GO ON
 9002 041714  104002                          ERROR   .+2                     ;MMU ERROR
 9003                                                                         ;NO GO TO ERROR
 9004 041716  005037  177776          402$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
 9005 041722  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
 9006 041726  001401                          BEQ     404$                    ;YES GO ON
 9007 041730  104002                          ERROR   .+2                     ;MMU ERROR
 9008                                                                         ;NO GO TO ERROR
 9009 041732  005046                  404$:   CLR     -(SP)                   ;SETUP STACK FOR NEXT TEST
 9010 041734  005237  177572                  INC     @#177572                ;TURN MMU ON
 9011 041740  006603                          MTPI    R3                      ; TEST INSTRUCTION
 9012 041742  022737  000004  177776          CMP     #4,@#177776             ;IS PSW CORRECT
 9013 041750  001401                          BEQ     5$                      ;YES GO ON
 9014 041752  104002                          ERROR   .+2                     ;MMU ERROR
 9015                                                                         ;NO GO TO ERROR
 9016 041754  005037  177572          5$:     CLR     @#177572                ;TURN MMU OFF
 9017 041760  020327  000000                  CMP     R3,#0                   ;IS R3 CORRECT
 9018 041764  001401                          BEQ     6$                      ;YES GO ON
 9019 041766  104002                          ERROR   .+2                     ;MMU ERROR
 9020                                                                         ;NO GO TO ERROR
 9021 041770  022627  177777          6$:     CMP     (SP)+,#177777           ;IS STACK CORRECT
 9022 041774  001401                          BEQ     7$                      ;YES GO ON
 9023 041776  104002                          ERROR   .+2                     ;MMU ERROR
 9024                                                                         ;NO GO TO ERROR
 9025 042000  012637  000244          7$:     MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
 9026
 9027                                 ;
 9030 042004                          TSMM6D:
 9031                                 ;       TEST MTPD (MOVE TO PREVIOUS DATA SPACE)
 9032 042004  005037  177572                  CLR     @#177572                ;MMU OFF
 9033 042010  005037  003030                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
 9034 042014  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
 9035 042022  010637  003040                  MOV     R6,@#SAVUSE             ;SAVE USER SP
```

MEMORY MANAGEMENT TESTS

```
 9036 042026  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 9037 042034  010637  003036                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
 9038 042040  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 9039 042046  004767  073014                  JSR     PC,MMU                  ;INIT MMU
 9040 042052  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
 9041 042060  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
 9042 042064  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON STACK
 9043 042070  012746  100004                  MOV     #100004,-(SP)           ;PUT TEST DATA ON STACK
 9044 042074  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
 9045 042102  012767  077400  135470          MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
 9046 042110  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
 9047 042114  005237  177572                  INC     @#177572                ;TURN MMU ON
 9048 042120  106623                          MTPD    (R3)+                   ; TEST INSTRUCTION
 9049 042122  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
 9050 042130  001401                          BEQ     1$                      ;YES GO ON
 9051 042132  104002                          ERROR   .+2             ;MMU ERROR
 9052                                         ;NO GO TO ERROR
 9053 042134  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
 9054 042140  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
 9055 042146  020637  003040                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
 9056 042152  001401                          BEQ     100$                    ;YES GO ON
 9057 042154  104002                          ERROR   .+2             ;MMU ERROR
 9058                                         ;NO GO TO ERROR
 9059 042156  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
 9060 042164  020637  003036                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
 9061 042170  001401                          BEQ     200$                    ;YES GO ON
 9062 042172  104002                          ERROR   .+2             ;MMU ERROR
 9063                                         ;NO GO TO ERROR
 9064 042174  023727  000244  100004  200$:   CMP     @#244,#100004           ;IS TEST LOCATION CORRECT
 9065 042202  001401                          BEQ     2$                      ;YES GO ON
 9066 042204  104002                          ERROR   .+2             ;MMU ERROR
 9067                                         ;NO GO TO ERROR
 9068 042206  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
 9069 042212  001401                          BEQ     3$                      ;YES GO ON
 9070 042214  104002                          ERROR   .+2             ;MMU ERROR
 9071                                         ;NO GO TO ERROR
 9072 042216  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
 9073 042222  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
 9074 042226  001401                          BEQ     4$                      ;YES GO ON
 9075 042230  104002                          ERROR   .+2             ;MMU ERROR
 9076                                         ;NO GO TO ERROR
 9077 042232  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
 9078 042240  012746  100737                  MOV     #100737,-(SP)           ;SETUP TEST DATA
 9079 042244  012701  000244                  MOV     #244,R1                 ;SETUP R1
 9080 042250  005237  177572                  INC     @#177572                ;TURN MMU ON
 9081 042254  106611                          MTPD    (R1)                    ;TEST INSTRUCTION
 9082 042256  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
 9083 042264  001401                          BEQ     300$                    ;YES GO ON
 9084 042266  104002                          ERROR   .+2             ;MMU ERROR
 9085                                         ;NO GO TO ERROR
 9086 042270  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
 9087 042274  023727  000244  100737          CMP     @#244,#100737           ;IS TEST LOCATION CORRECT
 9088 042302  001401                          BEQ     301$                    ;YES GO ON
 9089 042304  104002                          ERROR   .+2             ;MMU ERROR
 9090                                         ;NO GO TO ERROR
 9091 042306  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
 9092 042312  001401                          BEQ     302$                    ;YES GO ON
```

MEMORY MANAGEMENT TESTS

```
 9093 042314  104002                          ERROR   ;2              ;MMU ERROR
 9094                                                                 ;NO GO TO ERROR
 9095 042316  005037  177776          302$:   CLR     @#177776        ;SET PSW TO KERNEL MODE
 9096 042322  021627  177777                  CMP     (SP),#177777    ;IS STACK CORRECT
 9097 042326  001401                          BEQ     304$            ;YES GO ON
 9098 042330  104002                          ERROR   ;2              ;MMU ERROR
 9099                                                                 ;NO GO TO ERROR
 9100 042332  012737  030017  177776  304$:   MOV     #30017,@#177776 ;SETUP PSW
 9101 042340  012746  156711                  MOV     #156711,-(SP)   ;SETUP TEST DATA
 9102 042344  012701  000246                  MOV     #246,R1         ;SETUP R1
 9103 042350  005237  177572                  INC     @#177572        ;TURN MMU ON
 9104 042354  106641                          MTPD    -(R1)           ;TEST INSTRUCTION
 9105 042356  022737  030011  177776          CMP     #30011,@#177776 ;IS PSW CORRECT
 9106 042364  001401                          BEQ     400$            ;YES GO ON
 9107 042366  104002                          ERROR   ;2              ;MMU ERROR
 9108                                                                 ;NO GO TO ERROR
 9109 042370  005037  177572          400$:   CLR     @#177572        ;TURN MMU OFF
 9110 042374  023727  000244  156711          CMP     @#244,#156711   ;IS TEST LOCATION CORRECT
 9111 042402  001401                          BEQ     401$            ;YES GO ON
 9112 042404  104002                          ERROR   ;2              ;MMU ERROR
 9113                                                                 ;NO GO TO ERROR
 9114 042406  020127  000244          401$:   CMP     R1,#244         ;IS R1 CORRECT
 9115 042412  001401                          BEQ     402$            ;YES GO ON
 9116 042414  104002                          ERROR   ;2              ;MMU ERROR
 9117                                                                 ;NO GO TO ERROR
 9118 042416  005037  177776          402$:   CLR     @#177776        ;SET PSW TO KERNEL MODE
 9119 042422  021627  177777                  CMP     (SP),#177777    ;IS STACK CORRECT
 9120 042426  001401                          BEQ     404$            ;YES GO ON
 9121 042430  104002                          ERROR   ;2              ;MMU ERROR
 9122                                                                 ;NO GO TO ERROR
 9123 042432  005046                  404$:   CLR     -(SP)           ;SETUP STACK FOR NEXT TEST
 9124 042434  005237  177572                  INC     @#177572        ;TURN MMU ON
 9125 042440  106603                          MTPD    R3              ; TEST INSTRUCTION
 9126 042442  022737  000004  177776          CMP     #4,@#177776     ;IS PSW CORRECT
 9127 042450  001401                          BEQ     5$              ;YES GO ON
 9128 042452  104002                          ERROR   ;2              ;MMU ERROR
 9129                                                                 ;NO GO TO ERROR
 9130 042454  005037  177572          5$:     CLR     @#177572        ;TURN MMU OFF
 9131 042460  020327  000000                  CMP     R3,#0           ;IS R3 CORRECT
 9132 042464  001401                          BEQ     6$              ;YES GO ON
 9133 042466  104002                          ERROR   ;2              ;MMU ERROR
 9134                                                                 ;NO GO TO ERROR
 9135 042470  022627  177777          6$:     CMP     (SP)+,#177777   ;IS STACK CORRECT
 9136 042474  001401                          BEQ     7$              ;YES GO ON
 9137 042476  104002                          ERROR   ;2              ;MMU ERROR
 9138                                                                 ;NO GO TO ERROR
 9139 042500  012637  000244          7$:     MOV     (SP)+,@#244     ;RESTORE TEST LOCATION
 9140
 9141                                 ;
 9144 042504                          TSMMU7:
 9145                                 ;       TEST NON-RESIDENT ABORT
 9146 042504  005037  177572                  CLR     @#177572        ;MMU OFF
 9147 042510  005067  140314                  CLR     FLAG            ;CLEAR MMU ABORT FLAG
 9148 042514  013746  000214                  MOV     @#214,-(SP)     ;SAVE DATA AT TEST LOCATIONS
 9149 042520  013746  000216                  MOV     @#216,-(SP)     ;
 9150 042524  005067  140312                  CLR     SAVMR0          ;CLEAR STATUS REGS SAVE AREAS
 9151 042530  005067  140310                  CLR     SAVMR1          ;
```

MEMORY MANAGEMENT TESTS

```
9152 042534  005067  140306                CLR     SAVMR2                  ;
9153 042540  004767  072322                JSR     PC,MMU                  ;INIT MMU
9154 042544  012737  030000  177776        MOV     #30000,@#177776         ;SETUP PSW
9155 042552  012702  000200                MOV     #200,R2                 ;
9156 042556  012737  077400  177600        MOV     #77400,@#177600         ;SETUP FOR AN ABORT
9157 042564  004767  000164                JSR     PC,TS7                  ;CAUSE AN ABORT TO OCCUR AND
9158                                                                        ;THEN CHECK IF ABORT FLAG REGISTERED
9159                                                                        ;THIS EVENT AND CHECK IF STATUS REGS
9160                                                                        ;CONTAINED EXPECTED VALUES.
9161                                                                        ;IF NO ABORT OCCURRED THEN GO TO ERROR
9162                                                                        ;OTHERWISE CONTINUE.
9163 042570  012737  077404  177600        MOV     #77404,@#177600         ;SETUP FOR AN ABORT
9164 042576  004767  000152                JSR     PC,TS7                  ;CAUSE AN ABORT TO OCCUR AND
9165                                                                        ;THEN CHECK IF ABORT FLAG REGISTERED
9166                                                                        ;THIS EVENT AND CHECK IF STATUS REGS
9167                                                                        ;CONTAINED EXPECTED VALUES.
9168                                                                        ;IF NO ABORT OCCURRED THEN GO TO ERROR
9169                                                                        ;OTHERWISE CONTINUE.
9170 042602  012701  000220                MOV     #220,R1                 ;
9171 042606  004767  072254                JSR     PC,MMU                  ;INIT MMU
9172 042612  005003                        CLR     R3                      ;SETUP MMR1 EXPECTED DATA
9173 042614  012767  000001  140206        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
9174 042622  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
9175 042630  012737  100000  177776        MOV     #100000,@#177776        ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9176 042636  012241                        MOV     (R2)+,-(R1)             ;CAUSE AN ABORT
9177 042640  004767  000220                JSR     PC,TSM7                 ;CHECK IF AN ABORT OCCURRED BY
9178                                                                        ;CHECKING ABORT FLAG AND STATUS REGS
9179                                                                        ;IF NO ABORT OCCURRED THEN GO TO ERROR
9180                                                                        ;OTHERWISE CONTINUE.
9181 042644  005067  140172                CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9182 042650  005067  140170                CLR     SAVMR1                  ;
9183 042654  005067  140166                CLR     SAVMR2                  ;
9184 042660  012703  000022                MOV     #22,R3                  ;SETUP MMR1 EXPECTED DATA
9185 042664  012767  000001  140136        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
9186 042672  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
9187 042700  012737  020000  177776        MOV     #20000,@#177776         ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9188 042706  006522                        MFPI    (R2)+                   ;CAUSE AN ABORT
9189 042710  004767  000150                JSR     PC,TSM7                 ;CHECK IF AN ABORT OCCURRED BY
9190                                                                        ;CHECKING ABORT FLAG AND STATUS REGS
9191                                                                        ;IF NO ABORT OCCURRED THEN GO TO ERROR
9192                                                                        ;OTHERWISE CONTINUE.
9193 042714  012737  030000  177776        MOV     #30000,@#177776         ;SETUP PSW
9194 042722  012737  077400  177600        MOV     #77400,@#177600         ;SETUP FOR AN ABORT
9195 042730  005037  177572                CLR     @#177572                ;MMU OFF
9196 042734  006522                        MFPI    (R2)+                   ;TRY TO CAUSE AN ABORT
9197 042736  012603                        MOV     (SP)+,R3                ;POP THE STACK
9198 042740  012637  000216                MOV     (SP)+,@#216             ;RESTORE DATA AT TEST LOCATIONS
9199 042744  012637  000214                MOV     (SP)+,@#214             ;
9200
9201 042750  000167  000154                JMP     TS7FIN
9202                                ;
9203                                ;ROUTINE TO CAUSE AND CHECK  NONRESIDENT ABORTS
9204                                ;
9205 042754  012767  000001  140046  TS7:   MOV     #1,FLAG                 ;SETUP FOR AN ABORT
9206 042762  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
9207 042770  010701                        MOV     R7,R1                   ;SAVE PC
9208 042772  006522                        MFPI    (R2)+                   ;CAUSE AN ABORT
```

MEMORY MANAGEMENT TESTS

```
 9209 042774  022767  000000  140026          CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9210 043002  001401                          BEQ     OK7                     ;IF YES GO ON
 9211 043004  104002                          ERROR   .+2             ;MMU ERROR
 9212                                         ;IF NO GO TO ERROR
 9213 043006  105067  140030          OK7:    CLRB    SAVMR0                  ;SETUP EXPECTED DATA
 9214 043012  022767  100000  140022          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
 9215 043020  001401                          BEQ     OKA7                    ;IF OK THEN CONTINUE
 9216 043022  104002                          ERROR   .+2             ;MMU ERROR
 9217                                         ;NOT OK THEN GO TO ERROR
 9218 043024  026727  140014  000022  OKA7:   CMP     SAVMR1,#22              ; TEST MMR1 FOR EXPECTED VALUE
 9219 043032  001401                          BEQ     OKAY7                   ;IF OK THEN CONTINUE
 9220 043034  104002                          ERROR   .+2             ;MMU ERROR
 9221                                         ;NOT OK THEN GO TO ERROR
 9222 043036  026701  140004          OKAY7:  CMP     SAVMR2,R1               ; TEST MMR2 FOR EXPECTED VALUE
 9223 043042  001401                          BEQ     OKAY7A                  ;IF OK THEN CONTINUE
 9224 043044  104002                          ERROR   .+2             ;MMU ERROR
 9225                                         ;NOT OK THEN GO TO ERROR
 9226 043046  005067  137770          OKAY7A: CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9227 043052  005067  137766                  CLR     SAVMR1                  ;
 9228 043056  005067  137764                  CLR     SAVMR2                  ;
 9229 043062  000207                          RTS     PC                      ;RETURN
 9230                                 ;
 9231                                 ;ROUTINE TO CHECK IF A NONRESIDENT ABORT OCCURRED
 9232                                 ;
 9233 043064  022767  000000  137736  TSM7:   CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9234 043072  001401                          BEQ     TSMA                    ;IF YES GO ON
 9235 043074  104002                          ERROR   .+2             ;MMU ERROR
 9236                                         ;IF NO THEN GO TO ERROR
 9237 043076  042737  040377  003042  TSMA:   BIC     #40377,@#SAVMR0         ;SETUP EXPECTED DATA
 9238 043104  022767  100000  137730          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
 9239 043112  001401                          BEQ     TSMB                    ;IF OK THEN CONTINUE
 9240 043114  104002                          ERROR   .+2             ;MMU ERROR
 9241                                         ;IF NO THEN GO TO ERROR
 9242 043116  020367  137722          TSMB:   CMP     R3,SAVMR1               ; TEST MMR1 FOR EXPECTED VALUE
 9243 043122  001401                          BEQ     TSMC                    ;IF OK THEN CONTINUE
 9244 043124  104002                          ERROR   .+2             ;MMU ERROR
 9245                                         ;IF NOT OK THEN GO TO ERROR
 9246 043126  000207                  TSMC:   RTS     PC                      ;RETURN
 9247                                 ;
 9248 043130  000240                  TS7FIN: NOP
 9251 043132                          TSMMU8:
 9252                                 ;       TEST READ ONLY ABORTS
 9253 043132  005037  177572                  CLR     @#177572                ;MMU OFF
 9254 043136  005067  137666                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9255 043142  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATIONS
 9256 043146  013746  000246                  MOV     @#246,-(SP)             ;
 9257 043152  005067  137664                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9258 043156  005067  137662                  CLR     SAVMR1                  ;
 9259 043162  005067  137660                  CLR     SAVMR2                  ;
 9260 043166  004767  071674                  JSR     PC,MMU                  ;INIT MMU
 9261 043172  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
 9262 043200  012702  000244                  MOV     #244,R2                 ;
 9263 043204  012737  077402  177600          MOV     #77402,@#177600         ;SETUP FOR AN ABORT
 9264 043212  012746  000246                  MOV     #246,-(SP)              ;PUSH DATA ONTO THE STACK
 9265 043216  012767  000001  137604          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9266 043224  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
 9267 043232  010701                          MOV     R7,R1                   ;SAVE PC
```

MEMORY MANAGEMENT TESTS

```
9268 043234  006622                          MTPI    (R2)+           ;CAUSE ABORT
9269 043236  022767  000000  137564          CMP     #0,FLAG         ;DID ABORT OCCUR
9270 043244  001401                          BEQ     1$              ;IF YES THEN GO ON
9271 043246  104002                          ERROR   .2              ;MMU ERROR
9272                                                                 ;IF NO THEN GO TO ERROR
9273 043250  105067  137566          1$:     CLRB    SAVMR0          ;SETUP EXPECTED DATA
9274 043254  022767  020000  137560          CMP     #20000,SAVMR0   ; TEST MMR0 FOR EXPECTED VALUE
9275 043262  001401                          BEQ     2$              ;IF OK THEN CONTINUE
9276 043264  104002                          ERROR   .2              ;MMU ERROR
9277                                                                 ;OTHERWISE GO TO ERROR
9278 043266  022767  011026  137550  2$:     CMP     #11026,SAVMR1   ; TEST MMR1 FOR EXPECTED VALUE
9279 043274  001401                          BEQ     3$              ;IF OK THEN CONTINUE
9280 043276  104002                          ERROR   .2              ;MMU ERROR
9281                                                                 ;OTHERWISE GO TO ERROR
9282 043300  020167  137542          3$:     CMP     R1,SAVMR2       ; TEST MMR2 FOR EXPECTED VALUE
9283 043304  001401                          BEQ     4$              ;IF OK THEN CONTINUE
9284 043306  104002                          ERROR   .2              ;MMU ERROR
9285                                                                 ;OTHERWISE GO TO ERROR
9286 043310  012737  030000  177776  4$:     MOV     #30000,@#177776 ;SETUP PSW
9287 043316  012746  000002                  MOV     #2,-(SP)        ;PUSH DATA ONTO STACK
9288 043322  006622                          MTPI    (R2)+           ;TRY TO CAUSE ABORT
9289 043324  012637  000246                  MOV     (SP)+,@#246     ;RESTORE DATA AT TEST LOCATIONS
9290 043330  012637  000244                  MOV     (SP)+,@#244     ;
9291
9294 043334                          TSMMU9:
9295                                 ;       TEST PAGE LENGTH ERROR ABORTS
9296 043334  005037  177572                  CLR     @#177572        ;MMU OFF
9297 043340  005067  137464                  CLR     FLAG            ;CLEAR MMU ABORT FLAG
9298 043344  005067  137472                  CLR     SAVMR0          ;CLEAR STATUS REGS SAVE AREAS
9299 043350  005067  137470                  CLR     SAVMR1          ;
9300 043354  005067  137466                  CLR     SAVMR2          ;
9301 043360  012737  030000  177776          MOV     #30000,@#177776 ;SETUP PSW
9302 043366  004767  071474                  JSR     PC,MMU          ;INIT MMU
9303 043372  012703  043654                  MOV     #PLF0,R3        ;LET R3, R1, AND R2 POINT TO THE
9304 043376  012701  043726                  MOV     #BN0,R1         ;UPWARD EXPANSION TABLES
9305 043402  012702  043776                  MOV     #ABORT0,R2      ;
9306 043406  012737  000026  172516          MOV     #26,@#172516    ;DISABLE USER DATA SPACE
9307 043414  004767  000050                  JSR     PC,TSM9         ;TURN MMU ON
9308                                                                 ;DO RELOCATIONS FOR THE DIFFERENT
9309                                                                 ;VALUES OF THE PAGE LENGTH FIELD AND
9310                                                                 ;BLOCK NUMBER.  IF AN ABORT OCCURS
9311                                                                 ;CHECK TO SEE IF IT WAS SUPPOSED TO,
9312                                                                 ;AND IF YES CHECK ABORT FLAG AND
9313                                                                 ;STATUS REGISTERS.
9314 043420  012703  044046                  MOV     #PLF1,R3        ;LET R3, R1, AND R2 POINT TO THE
9315 043424  012701  044116                  MOV     #BN1,R1         ;DOWNWARD EXPANSION TABLES
9316 043430  012702  044164                  MOV     #ABORT7,R2      ;
9317 043434  004767  000030                  JSR     PC,TSM9         ;TURN MMU ON
9318                                                                 ;DO RELOCATIONS FOR THE DIFFERENT
9319                                                                 ;VALUES OF THE PAGE LENGTH FIELD AND
9320                                                                 ;BLOCK NUMBER.  IF AN ABORT OCCURS
9321                                                                 ;CHECK TO SEE IF IT WAS SUPPOSED TO,
9322                                                                 ;AND IF YES CHECK ABORT FLAG AND
9323                                                                 ;STATUS REGISTERS.
9324 043440  005037  177572                  CLR     @#177572        ;MMU OFF
9325 043444  012703  044056                  MOV     #PLF1+10,R3     ;POINT TO A VALUE WHICH SHOULD CAUSE
9326 043450  012701  044126                  MOV     #BN1+10,R1      ;AN ABORT IF MMU IS ON.
```

```
043454  011337  177600                  MOV     (R3),@#177600           ;SETUP UIPDR0
043460  006521                          MFPI    (R1)+                   ;DO A RELOCATION
043462  012605                          MOV     (SP)+,R5                ;POP THE STACK

043464  000167  000542                  JMP     TS9FIN
                                ;
                                ;ROUTINE TO CAUSE AND CHECK PAGE LENGTH ERROR ABORTS
                                ;
043470  012337  177600          TSM9:   MOV     (R3)+,@#177600          ;SETUP UIPDR0
043474  010100                          MOV     R1,R0                   ;SAVE A COPY OF R1
043476  012767  000001  137324          MOV     #1,FLAG                 ;SETUP FOR AN ABORT
043504  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
043512  010704                          MOV     R7,R4                   ;SAVE PC
043514  006530                          MFPI    @(R0)+                  ;DO A RELOCATION OPERATION
043516  021227  000000                  CMP     (R2),#0                 ;WAS AN ABORT SUPPOSED TO OCCUR
043522  001007                          BNE     2$                      ;IF YES GO TO 2$
043524  012605                          MOV     (SP)+,R5                ;POP THE STACK
043526  022767  000001  137274          CMP     #1,FLAG                 ;DID AN ABORT OCCUR
043534  001401                          BEQ     1$                      ;NO GO ON
043536  104002                          ERROR   +2                  ;MMU ERROR
                                                                    ;YES GO TO ERROR
043540  000425                  1$:     BR      6$                      ;
043542  022767  000000  137260  2$:     CMP     #0,FLAG                 ;DID AN ABORT OCCUR
043550  001401                          BEQ     3$                      ;YES GO ON
043552  104002                          ERROR   +2                  ;MMU ERROR
                                                                    ;NO GO TO ERROR
043554  105067  137262          3$:     CLRB    SAVMR0                  ;SETUP EXPECTED DATA
043560  022767  040000  137254          CMP     #40000,SAVMR0           ; TEST MMR0 FOR EXPECTED VALUE
043566  001401                          BEQ     4$                      ;IF OK THEN CONTINUE
043570  104002                          ERROR   +2                  ;MMU ERROR
                                                                    ;NOT OK THEN GO TO ERROR
043572  022767  000020  137244  4$:     CMP     #20,SAVMR1              ; TEST MMR1 FOR EXPECTED VALUE
043600  001401                          BEQ     5$                      ;IF OK THEN CONTINUE
043602  104002                          ERROR   +2                  ;MMU ERROR
                                                                    ;NOT OK THEN GO TO ERROR
043604  020467  137236          5$:     CMP     R4,SAVMR2               ; TEST MMR2 FOR EXPECTED VALUE
043610  001401                          BEQ     6$                      ;IF OK THEN CONTINUE
043612  104002                          ERROR   +2                  ;MMU ERROR
                                                                    ;NOT OK THEN GO TO ERROR
043614  005067  137210          6$:     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
043620  005067  137216                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
043624  005067  137214                  CLR     SAVMR1                  ;
043630  005067  137212                  CLR     SAVMR2                  ;
043634  005201                          INC     R1                      ;POINT TO NEXT ENTRY
043636  005201                          INC     R1                      ;
043640  005202                          INC     R2                      ;
043642  005202                          INC     R2                      ;
043644  021327  000777                  CMP     (R3),#777               ;HAVE ALL ENTRIES BEEN TRIED
043650  001307                          BNE     TSM9                    ;NO REPEAT
043652  000207                          RTS     PC                      ;YES RETURN
                                ;
                                ;UPWARD EXPANSION TABLES
                                ;
043654  070006                  PLF0:   .WORD   70006
043656  070006                          .WORD   70006
043660  070006                          .WORD   70006
043662  013406                          .WORD   13406
```

MEMORY MANAGEMENT TESTS

```
9384 043664  020006                        .WORD   20006
9385 043666  004006                        .WORD   04006
9386 043670  040006                        .WORD   40006
9387 043672  070006                        .WORD   70006
9388 043674  024006                        .WORD   24006
9389 043676  004006                        .WORD   04006
9390 043700  014006                        .WORD   14006
9391 043702  012006                        .WORD   12006
9392 043704  002006                        .WORD   02006
9393 043706  001406                        .WORD   01406
9394 043710  004006                        .WORD   04006
9395 043712  002006                        .WORD   02006
9396 043714  000406                        .WORD   00406
9397 043716  007406                        .WORD   07406
9398 043720  001006                        .WORD   01006
9399 043722  003406                        .WORD   03406
9400 043724  000777                        .WORD   777
9401 043726  013000                BNO:    .WORD   013000
9402 043730  016000                        .WORD   016000
9403 043732  017000                        .WORD   017000
9404 043734  002700                        .WORD   002700
9405 043736  014000                        .WORD   014000
9406 043740  002000                        .WORD   002000
9407 043742  004000                        .WORD   004000
9408 043744  007000                        .WORD   007000
9409 043746  002000                        .WORD   002000
9410 043750  000700                        .WORD   000700
9411 043752  004000                        .WORD   004000
9412 043754  001000                        .WORD   001000
9413 043756  000300                        .WORD   000300
9414 043760  000400                        .WORD   000400
9415 043762  001400                        .WORD   001400
9416 043764  000600                        .WORD   000600
9417 043766  000200                        .WORD   000200
9418 043770  001700                        .WORD   001700
9419 043772  000300                        .WORD   000300
9420 043774  000700                        .WORD   000700
9421 043776  000000                ABORTO: .WORD   0
9422 044000  000000                        .WORD   0
9423 044002  000001                        .WORD   1
9424 044004  000000                        .WORD   0
9425 044006  000001                        .WORD   1
9426 044010  000001                        .WORD   1
9427 044012  000000                        .WORD   0
9428 044014  000000                        .WORD   0
9429 044016  000000                        .WORD   0
9430 044020  000000                        .WORD   0
9431 044022  000001                        .WORD   1
9432 044024  000000                        .WORD   0
9433 044026  000000                        .WORD   0
9434 044030  000001                        .WORD   1
9435 044032  000001                        .WORD   1
9436 044034  000001                        .WORD   1
9437 044036  000001                        .WORD   1
9438 044040  000000                        .WORD   0
9439 044042  000001                        .WORD   1
9440 044044  000000                        .WORD   0
```

```
                                        ;
                                        ;DOWNWARD EXPANSION TABLES
                                        ;
9444 044046  000416             PLF1:   .WORD   00416
9445 044050  020016                     .WORD   20016
9446 044052  024016                     .WORD   24016
9447 044054  034016                     .WORD   34016
9448 044056  074016                     .WORD   74016
9449 044060  040016                     .WORD   40016
9450 044062  020016                     .WORD   20016
9451 044064  000016                     .WORD   00016
9452 044066  030016                     .WORD   30016
9453 044070  010016                     .WORD   10016
9454 044072  014016                     .WORD   14016
9455 044074  004016                     .WORD   04016
9456 044076  002016                     .WORD   02016
9457 044100  000416                     .WORD   00416
9458 044102  000016                     .WORD   00016
9459 044104  003416                     .WORD   03416
9460 044106  001016                     .WORD   01016
9461 044110  001416                     .WORD   01416
9462 044112  000416                     .WORD   00416
9463 044114  000777                     .WORD   777
9464 044116  000100             BN1:    .WORD   000100
9465 044120  010000                     .WORD   010000
9466 044122  006000                     .WORD   006000
9467 044124  016000                     .WORD   016000
9468 044126  016000                     .WORD   016000
9469 044130  004000                     .WORD   004000
9470 044132  000000                     .WORD   000000
9471 044134  000000                     .WORD   000000
9472 044136  004000                     .WORD   004000
9473 044140  004000                     .WORD   004000
9474 044142  004000                     .WORD   004000
9475 044144  000000                     .WORD   000000
9476 044146  000300                     .WORD   000300
9477 044150  000000                     .WORD   000000
9478 044152  000400                     .WORD   000400
9479 044154  001000                     .WORD   001000
9480 044156  000100                     .WORD   000100
9481 044160  000400                     .WORD   000400
9482 044162  000200                     .WORD   000200
9483 044164  000000             ABORT7: .WORD   0
9484 044166  000000                     .WORD   0
9485 044170  000000                     .WORD   0
9486 044172  000000                     .WORD   0
9487 044174  000001                     .WORD   1
9488 044176  000001                     .WORD   1
9489 044200  000001                     .WORD   1
9490 044202  000000                     .WORD   0
9491 044204  000001                     .WORD   1
9492 044206  000000                     .WORD   0
9493 044210  000000                     .WORD   0
9494 044212  000001                     .WORD   1
9495 044214  000001                     .WORD   1
9496 044216  000001                     .WORD   1
9497 044220  000000                     .WORD   0
```

```
 9498 044222  000000                          .WORD   0
 9499 044224  000001                          .WORD   1
 9500 044226  000000                          .WORD   0
 9501 044230  000000                          .WORD   0
 9502                                ;
 9503 044232  000240                TS9FIN: NOP
 9506 044234                        TSMM10:
 9507                                ;       FUNCTIONAL TEST OF BITS <6:1> OF MMR0
 9508 044234  005037  177572                CLR     @#177572                ;MMU OFF
 9509 044240  005067  136564                CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9510 044244  005067  136572                CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9511 044250  005067  136570                CLR     SAVMR1                  ;
 9512 044254  005067  136566                CLR     SAVMR2                  ;
 9513 044260  004767  070602                JSR     PC,MMU                  ;INIT MMU
 9514 044264  005037  177776                CLR     @#177776                ;INIT PSW: PREVIOUS MODE = KERNAL
 9515 044270  012702  020200                MOV     #20200,R2               ;
 9516 044274  012737  077400  172302        MOV     #77400,@#172302         ;SETUP KIPDR1 TO ABORT
 9517 044302  012767  000001  136520        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9518 044310  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
 9519 044316  010701                        MOV     R7,R1                   ;SAVE PC
 9520 044320  006522                        MFPI    (R2)+                   ;DO A RELOCATION VIA KIPAR1
 9521 044322  012704  100003                MOV     #100003,R4              ;SETUP EXPECTED DATA
 9522 044326  004767  000202                JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9523                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.
 9524 044332  012737  030000  177776        MOV     #30000,@#177776         ;INIT PSW: PREVIOUS MODE = USER
 9525 044340  004767  070522                JSR     PC,MMU                  ;INIT MMU
 9526 044344  012737  077400  177636        MOV     #77400,@#177636         ;SETUP UDPDR7 TO ABORT
 9527 044352  012702  160000                MOV     #160000,R2              ;
 9528 044356  012767  000001  136444        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9529 044364  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
 9530 044372  010701                        MOV     R7,R1                   ;SAVE PC
 9531 044374  106522                        MFPD    (R2)+                   ;DO A RELOCATION VIA UDPAR7
 9532 044376  012704  100177                MOV     #100177,R4              ;SETUP EXPECTED DATA
 9533 044402  004767  000126                JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9534                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.
 9535 044406  012737  010000  177776        MOV     #10000,@#177776         ;INIT PSW: PREVIOUS MODE= SUPERVISOR
 9536 044414  004767  070446                JSR     PC,MMU                  ;INIT MMU
 9537 044420  012737  077400  172212        MOV     #77400,@#172212         ;SETUP SIPDR5 TO ABORT
 9538 044426  012702  120000                MOV     #120000,R2              ;ACCESS PAGE 05
 9539 044432  012767  000001  136370        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9540 044440  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
 9541 044446  010701                        MOV     R7,R1                   ;SAVE PC
 9542 044450  006522                        MFPI    (R2)+                   ;DO A RELOCATION VIA SIPAR5
 9543 044452  012704  100053                MOV     #100053,R4              ;SETUP EXPECTED DATA:ABORT, PAGE 05
 9544 044456  004767  000052                JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
 9545                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.
 9546                                ;
 9547                                ;TEST THAT ILLEGAL MODE CAUSES MMU ABORT
 9548                                ;
 9549 044462  012737  020000  177776        MOV     #20000,@#177776         ;INIT PSW:SET ILLEGAL PREVIOUS MODE
 9550 044470  004767  070372                JSR     PC,MMU                  ;INIT MMU
 9551 044474  012702  040000                MOV     #40000,R2               ;SET UP ACCESS TO PAGE 2
 9552 044500  012767  000001  136322        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9553 044506  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
 9554 044514  010701                        MOV     R7,R1                   ;SAVE PC
 9555 044516  106522                        MFPD    (R2)+                   ;DO A RELOCATION
 9556 044520  012704  100105                MOV     #100105,R4              ;SETUP EXPECTED DATA:ABORT, ILLEGAL
```

```
                                                                        ; PROCESSOR MODE, PAGE 02
044524  004767  000004                JSR     PC,TS10                   ;CHECK IF AN ABORT OCCURRED AND
                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.

044530  000167  000062                JMP     T10FIN
                              ;
                              ;ROUTINE TO CHECK IF A MMU ABORT OCCURRED AND IF STATUS REG MMR0
                              ;CONTAINS EXPECTED DATA
                              ;
044534  022767  000000  136266 TS10:   CMP     #0,FLAG                   ;DID AN ABORT OCCUR
044542  001401                        BEQ     1$                        ;YES GO ON
044544  104002                        ERROR   .+2                   ;MMU ERROR
                                                                    ;NO GO TO ERROR
044546  020467  136270        1$:     CMP     R4,SAVMR0                 ; TEST MMR0 FOR EXPECTED DATA
044552  001401                        BEQ     2$                        ;OK GO ON
044554  104002                        ERROR   .+2                   ;MMU ERROR
                                                                    ;NO GO TO ERROR
044556  022767  000022  136260 2$:     CMP     #22,SAVMR1                ; TEST MMR1 FOR EXPECTED DATA
044564  001401                        BEQ     3$                        ;OK GO ON
044566  104002                        ERROR   .+2                   ;MMU ERROR
                                                                    ;NO GO TO ERROR
044570  020167  136252        3$:     CMP     R1,SAVMR2                 ; TEST MMR2 FOR EXPECTED DATA
044574  001401                        BEQ     4$                        ;OK GO ON
044576  104002                        ERROR   .+2                   ;MMU ERROR
                                                                    ;NO GO TO ERROR
044600  005067  136236        4$:     CLR     SAVMR0                    ;CLEAR MMU STATUS REGS SAVE AREAS
044604  005067  136234                CLR     SAVMR1                    ;
044610  005067  136232                CLR     SAVMR2                    ;
044614  000207                        RTS     PC                        ;RETURN
                              ;
044616                        T10FIN:
044616                        TSMM11:
                              ;       TEST DATA SPACE BITS MMR3
044616  005037  177572                CLR     @#177572                  ;MMU OFF
044622  005067  136202                CLR     FLAG                      ;CLEAR MMU ABORT FLAG
044626  012737  030000  177776        MOV     #30000,@#177776           ;SETUP PSW
044634  012701  000026                MOV     #26,R1                    ;SETUP FIRST MMR3 VALUE
044640  012703  177610                MOV     #177610,R3                ;POINT TO UIPDR4
044644  012704  000021                MOV     #21,R4                    ;SETUP SECOND MMR3 VALUE
044650  004767  000060                JSR     PC,TS11                   ; TEST ENABLE USER DATA SPACE BIT
044654  012737  000000  177776        MOV     #0,@#177776               ;SETUP PSW
044662  012701  000023                MOV     #23,R1                    ;SETUP FIRST MMR3 VALUE
044666  012703  172310                MOV     #172310,R3                ;POINT TO KIPDR4
044672  012704  000024                MOV     #24,R4                    ;SETUP SECOND MMR3 VALUE
044676  004767  000032                JSR     PC,TS11                   ; TEST ENABLE KERNEL DATA SPACE BIT
044702  012737  010000  177776        MOV     #10000,@#177776           ;SETUP PSW
044710  012701  000025                MOV     #25,R1                    ;SETUP FIRST MMR3 VALUE
044714  012703  172210                MOV     #172210,R3                ;POINT TO SIPDR4
044720  012704  000022                MOV     #22,R4                    ;SETUP SECOND MMR3 VALUE
044724  004767  000004                JSR     PC,TS11                   ; TEST ENABLE SUPERVISOR DATA SPACE BIT

044730  000167  000120                JMP     T11FIN
                              ;
                              ;ROUTINE TO TEST ENABLE DATA SPACE BITS OF MMR3
                              ;
044734  004767  070126        TS11:   JSR     PC,MMU                    ;INIT MMU
044740  010137  172516                MOV     R1,@#172516               ;DISABLE DATA SPACE OF MODE UNDER TEST
```

```
044744  012713  077400                  MOV     #77400,(R3)             ;SETUP IPDR TO ABORT
044750  012702  100000                  MOV     #100000,R2              ;
044754  012767  000001  136046          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
044762  012737  000001  177572          MOV     #1,@#177572             ;MMU ON
044770  106522                          MFPD    (R2)+                   ;DO A RELOCATION
044772  022767  000000  136030          CMP     #0,FLAG                 ;DID AN ABORT OCCUR
045000  001401                          BEQ     1$                      ;YES GO ON
045002  104002                          ERROR   +2                ;MMU ERROR
                                                                  ;NO GO TO ERROR
045004  010437  172516          1$:     MOV     R4,@#172516             ;ENABLE DATA SPACE OF MODE UNDER TEST
045010  012702  100000                  MOV     #100000,R2              ;
045014  012767  000001  136006          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
045022  012737  000001  177572          MOV     #1,@#177572             ;MMU ON
045030  106522                          MFPD    (R2)+                   ;DO A RELOCATION
045032  005726                          TST     (SP)+                   ;POP THE STACK
045034  022767  000001  135766          CMP     #1,FLAG                 ;DID AN ABORT OCCUR
045042  001401                          BEQ     2$                      ;NO GO ON
045044  104002                          ERROR   +2                ;MMU ERROR
                                                                  ;YES GO TO ERROR
045046  005067  135756          2$:     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
045052  000207                          RTS     PC                      ;RETURN
                                ;
045054                          T11FIN:
045054                          TSMM12:
                                ;       MMR1 FUNCTIONAL TEST
045054  005037  177572                  CLR     @#177572                ;MMU OFF
045060  005067  135744                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
045064  005067  135754                  CLR     SAVMR1                  ;CLEAR STATUS REG SAVE AREA
045070  004767  067772                  JSR     PC,MMU                  ;INIT MMU
045074  012737  030000  177776          MOV     #30000,@#177776         ;INIT PSW
045102  012704  100200                  MOV     #100200,R4              ;SETUP TEST LOCATIONS
045106  010401                          MOV     R4,R1                   ;
045110  012705  100101                  MOV     #100101,R5              ;
045114  010502                          MOV     R5,R2                   ;
045116  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
045124  012737  077402  172310          MOV     #77402,@#172310         ;SETUP KIPDR4 TO ABORT
045132  012703  006414                  MOV     #6414,R3                ;SETUP EXPECTED DATA FOR MMR1
045136  012767  000001  135664          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
045144  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
Z 045152  010767  135634                  MOV     R7,SLOC00               ;SAVE PC
045156  112425                          MOVB    (R4)+,(R5)+             ;DO A RELOCATION
045160  004767  000206                  JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
                                                                        ;YES IF MMR1 EQUALS EXPECTED DATA
045164  012703  175011                  MOV     #175011,R3              ;SETUP EXPECTED DATA FOR MMR1
045170  012767  000001  135632          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
045176  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
Z 045204  010767  135602                  MOV     R7,SLOC00               ;SAVE PC
045210  112142                          MOVB    (R1)+,-(R2)             ;DO A RELOCATION
045212  004767  000154                  JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
                                                                        ;YES IF MMR1 EQUALS EXPECTED DATA
045216  012703  006771                  MOV     #6771,R3                ;SETUP EXPECTED DATA FOR MMR1
045222  012767  000001  135600          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
045230  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
Z 045236  010767  135550                  MOV     R7,SLOC00               ;SAVE PC
045242  114125                          MOVB    -(R1),(R5)+             ;DO A RELOCATION
045244  004767  000122                  JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
                                                                        ;YES IF MMR1 EQUALS EXPECTED DATA
```

MEMORY MANAGEMENT TESTS

```
    9675 045250  012703  006411                MOV     #6411,R3                ;SETUP EXPECTED DATA FOR MMR1
    9676 045254  012767  000001  135546        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
    9677 045262  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
Z   9678 045270  010767  135516                MOV     R7,SLOC00               ;SAVE PC
    9679 045274  112125                        MOVB    (R1)+,(R5)+             ;DO A RELOCATION
    9680 045276  004767  000070                JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
    9681                                                                       ;YES IF MMR1 EQUALS EXPECTED DATA
    9682 045302  012703  171025                MOV     #171025,R3              ;SETUP EXPECTED DATA FOR MMR1
    9683 045306  012767  000001  135514        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
    9684 045314  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
Z   9685 045322  010767  135464                MOV     R7,SLOC00               ;SAVE PC
    9686 045326  012542                        MOV     (R5)+,-(R2)             ;DO A RELOCATION
    9687 045330  004767  000036                JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
    9688                                                                       ;YES IF MMR1 EQUALS EXPECTED DATA
    9689 045334  012703  012762                MOV     #12762,R3               ;SETUP EXPECTED DATA FOR MMR1
    9690 045340  012767  000001  135462        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
    9691 045346  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
Z   9692 045354  010767  135432                MOV     R7,SLOC00               ;SAVE PC
    9693 045360  014225                        MOV     -(R2),(R5)+             ;DO A RELOCATION
    9694 045362  004767  000004                JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
    9695                                                                       ;YES IF MMR1 EQUALS EXPECTED DATA
    9696
    9697 045366  000167  000046                JMP     T12FIN
    9698                                ;
    9699                                ;ROUTINE TO CHECK IF AN ABORT OCCURRED AND IF MMR1 EQUALS EXPECTED DATA
    9700                                ;
    9701 045372  022767  000000  135430  TS12:  CMP     #0,FLAG                 ;DID AN ABORT OCCUR
    9702 045400  001401                        BEQ     1$                      ;YES GO ON
    9703 045402  104002                        ERROR   .+2                     ;MMU ERROR
    9704                                                                       ;NO GO TO ERROR
    9705 045404  020367  135434          1$:   CMP     R3,SAVMR1                       ; TEST MMR1 FOR EXPECTED DATA
    9706 045410  001401                        BEQ     2$                      ;OK GO ON
    9707 045412  104002                        ERROR   .+2                     ;MMU ERROR
    9708                                                                       ;NO GO TO ERROR
    9709 045414  026767  135372  135424  2$:   CMP     SLOC00,SAVMR2           ; TEST MMR2 FOR EXPECTED DATA
    9710 045422  001401                        BEQ     3$                      ;OK GO ON
    9711 045424  104002                        ERROR   .+2                     ;MMU ERROR
    9712                                                                       ;NO GO TO ERROR
    9713 045426  005067  135412          3$:   CLR     SAVMR1                  ;CLEAR STATUS REG SAVE AREA
    9714 045432  005067  135410                CLR     SAVMR2                  ;
    9715 045436  000207                        RTS     PC                      ;RETURN
    9716                                ;
    9717 045440  000240                T12FIN: NOP
    9727 045442                        TSMM13:
    9728                                ;       ADDER RELOCATION TEST PART A
    9729                                ;;**************************************************************************
                                        ;(NEED 16 BITS OF MEMORY ADDRESSING)
    9730 045442  005037  177572                CLR     @#177572                ;MMU OFF
    9731 045446  005067  135356                CLR     FLAG                    ;CLEAR MMU ABORT FLAG
    9732 045452  005037  177776                CLR     @#177776                ;INIT PSW
    9733 045456  004767  067404                JSR     PC,MMU                  ;INIT MMU
    9734 045462  012737  000020  172516        MOV     #20,@#172516            ;INIT MMR3
    9735 045470  012703  045644                MOV     #PARAD1,R3              ;SETUP PARS WITH TEST VALUES
    9736 045474  012701  045676                MOV     #PARVA1,R1              ;
    9737 045500  012133                1$:     MOV     (R1)+,@(R3)+            ;
    9738 045502  021127  000333                CMP     (R1),#333               ;
    9739 045506  001374                        BNE     1$                      ;
```

MEMORY MANAGEMENT TESTS

```
9740 045510  012703  045762              MOV     #PHY1,R3              ;SET POINTERS TO ADDER PART A
9741 045514  012701  045730              MOV     #VIR1,R1              ; TEST TABLES.
9742 045520  012702  046014              MOV     #MODE1,R2             ;
9743 045524  012237  177776      2$:     MOV     (R2)+,@#177776        ;INIT PSW
9744 045530  013305                      MOV     @(R3)+,R5             ;SAVE DATA AT PHYSICAL ADDRESS
9745 045532  012737  000001  177572      MOV     #1,@#177572           ;TURN MMU ON
9746 045540  006531                      MFPI    @(R1)+                ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9747 045542  012604                      MOV     (SP)+,R4              ;
9748 045544  005037  177572              CLR     @#177572              ;TURN MMU OFF
9749 045550  020504                      CMP     R5,R4                 ;IS DATA EQUAL TO EXPECTED
9750 045552  001401                      BEQ     3$                    ;YES GO ON
9751 045554  104002                      ERROR   +2                ;MMU ERROR
9752                                                                   ;NO IT IS AN ADDER ERROR
9753 045556  021327  000111      3$:     CMP     (R3),#111             ;ARE WE READY TO TEST DATA SPACE
9754 045562  001360                      BNE     2$                    ;NO GO TO 2$
9755 045564  005203                      INC     R3                    ;POINT TO DATA SPACE VALUES
9756 045566  005203                      INC     R3                    ;
9757 045570  005201                      INC     R1                    ;
9758 045572  005201                      INC     R1                    ;
9759 045574  005202                      INC     R2                    ;
9760 045576  005202                      INC     R2                    ;
9761 045600  012237  177776              MOV     (R2)+,@#177776        ;INIT PSW
9762 045604  013305                      MOV     @(R3)+,R5             ;SAVE DATA AT PHYSICAL ADDRESS
9763 045606  012737  000027  172516      MOV     #27,@#172516          ;INIT MMR3
9764 045614  012737  000001  177572      MOV     #1,@#177572           ;TURN MMU ON
9765 045622  106531                      MFPD    @(R1)+                ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9766 045624  012604                      MOV     (SP)+,R4              ;POP THE STACK
9767 045626  005037  177572              CLR     @#177572              ;TURN MMU OFF
9768 045632  020504                      CMP     R5,R4                 ;IS DATA EQUAL TO EXPECTED
9769 045634  001401                      BEQ     4$                    ;YES GO ON
9770 045636  104002                      ERROR   +2                ;MMU ERROR
9771                                                                   ;NO IT IS AN ADDER ERROR
9772 045640                      4$:
9773 045640  000167  000202              JMP     T13FIN
9774                             ;
9775                             ;ADDER TEST PART A TABLES
9776                             ;
9777 045644  172240      PARAD1: .WORD   172240
9778 045646  177642              .WORD   177642
9779 045650  172252              .WORD   172252
9780 045652  177640              .WORD   177640
9781 045654  172242              .WORD   172242
9782 045656  172254              .WORD   172254
9783 045660  177652              .WORD   177652
9784 045662  177644              .WORD   177644
9785 045664  172246              .WORD   172246
9786 045666  177654              .WORD   177654
9787 045670  172250              .WORD   172250
9788 045672  177660              .WORD   177660
9789 045674  000333              .WORD   333
9790 045676  000000      PARVA1: .WORD   000000
9791 045700  000010              .WORD   000010
9792 045702  177777              .WORD   177777
9793 045704  177601              .WORD   177601
9794 045706  000010              .WORD   000010
9795 045710  000052              .WORD   000052
9796 045712  000070              .WORD   000070
```

MEMORY MANAGEMENT TESTS

```
 9797 045714  000010                          .WORD   000010
 9798 045716  000010                          .WORD   000010
 9799 045720  000060                          .WORD   000060
 9800 045722  000000                          .WORD   000000
 9801 045724  000010                          .WORD   000010
 9802 045726  000333                          .WORD   333
 9803 045730  000000                VIR1:     .WORD   000000
 9804 045732  025000                          .WORD   025000
 9805 045734  135224                          .WORD   135224
 9806 045736  017700                          .WORD   017700
 9807 045740  033000                          .WORD   033000
 9808 045742  145252                          .WORD   145252
 9809 045744  121000                          .WORD   121000
 9810 045746  043000                          .WORD   043000
 9811 045750  075000                          .WORD   075000
 9812 045752  142000                          .WORD   142000
 9813 045754  117700                          .WORD   117700
 9814 045756  000111                          .WORD   111
 9815 045760  007000                          .WORD   007000
 9816 045762  000000                PHY1:     .WORD   000000
 9817 045764  006000                          .WORD   006000
 9818 045766  015124                          .WORD   015124
 9819 045770  000000                          .WORD   000000
 9820 045772  014000                          .WORD   014000
 9821 045774  012452                          .WORD   012452
 9822 045776  010000                          .WORD   010000
 9823 046000  004000                          .WORD   004000
 9824 046002  016000                          .WORD   016000
 9825 046004  010000                          .WORD   010000
 9826 046006  017700                          .WORD   017700
 9827 046010  000111                          .WORD   111
 9828 046012  010000                          .WORD   010000
 9829 046014  010000                MODE1:    .WORD   010000
 9830 046016  030000                          .WORD   030000
 9831 046020  010000                          .WORD   010000
 9832 046022  030000                          .WORD   030000
 9833 046024  010000                          .WORD   010000
 9834 046026  010000                          .WORD   010000
 9835 046030  030000                          .WORD   030000
 9836 046032  030000                          .WORD   030000
 9837 046034  010000                          .WORD   010000
 9838 046036  030000                          .WORD   030000
 9839 046040  010000                          .WORD   010000
 9840 045042  000111                          .WORD   111
 9841 046044  030000                          .WORD   030000
 9842                                ;
 9843 046046                         T13FIN:
 9854 046046                         TS1822:
 9855                                ;       TEST 22/18 BIT ADDRESS OPTION
 9856                                ;;*************************************************************************
                                     ;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.
 9857 046046  032777  000400  133064          BIT     #BIT08,@SWR             ;IS BIT 08 SET?
 9858 046054  001405                          BEQ     100$                    ;BRANCH IF ITS NOT
 9859 046056  062737  000001  001204          ADD     #1,@#TESTN              ;KEEP TEST NUMBERS IN ORDER
 9860                                                                         ;ADD 1 FOR THE TESTS WE'RE SKIPPING
```

MEMORY MANAGEMENT TESTS

```
 9861 046064  000167  001422                    JMP     T14FIN                  ;SKIP OVER THESE TESTS IF IT IS
 9862                                 ;
 9863                                 ;IF THIS IS A 22 BIT SYSTEM CHECK THE 22/18 BIT ADDRESS OPTION
 9864                                 ;
 9865                                 ;TO TEST 22 BIT ADDRESSING WE DO THE FOLLOWING:
 9866                                 ;       A. ENABLE 22 BIT ADDRESSING MODE
 9867                                 ;       B. CLEAR ADDRESS 0
 9868                                 ;       C. WRITE PHYSICAL ADDRESS 17000000 WITH ALL ONES
 9869                                 ;       D. CHECK ADDRESS 0
 9870                                 ;IF ADDRESS 0 IS UNCHANGED (=0) OR A TIME OUT OCCURRED, IT INDICATES
 9871                                 ;22 BIT MODE IS FUNCTIONING.
 9872                                 ;IF ADDRESS 0 =177777 IT INDICATES THAT 22 BIT MODE IS NOT FUNCTIONING.
 9873
 9874 046070  013767  000004  134714  100$:   MOV     @#4,SLOC00              ;SAVE VECTOR
 9875 046076  013767  000006  134710          MOV     @#6,SLOC01              ;SAVE VECTOR
 9876 046104  012737  046164  000004          MOV     #1$,@#4                 ;SET VECTOR FOR NXM TRAP
 9877 046112  012737  000340  000006          MOV     #340,@#6                ;
 9878 046120  012767  000020  124370          MOV     #20,SR3                 ;ENABLE 22 BIT MODE ADDRESSING
 9879 046126  012767  170000  124220          MOV     #170000,KIPAR6          ;SET KIPAR6 FOR 1920-1924KW ADDR RANGE
 9880 046134  012767  000001  131430          MOV     #1,SR0                  ;ENABLE MMU
 9881 046142  005067  131632                  CLR     0                       ;CLEAR ADDR 0
 9882 046146  012737  177777  140000          MOV     #177777,@#140000        ;MOVE ALL ONES TO ADDR 17000000 VIA KIPAR6
 9883                                                                         ;A TIME OUT ERROR OR
 9884 046154  005767  131620                  TST     0                       ;ADDR 0 REMAINING CLEAR INDICATES
 9885                                                                         ;THAT 22 BIT ADDRESS MODE IS WORKING AND
 9886                                                                         ;THAT SOME FURTHER TESTS SHOULD BE PERFORMED
 9887 046160  001405                          BEQ     2$                      ;IF ADDR 0 =177777
 9888                                                                         ;ERROR! 22 BIT ADDRESS MODE BAD
 9889 046162  104002                          ERROR   +2              ;MMU ERROR
 9890 046164  012706  001000          1$:     MOV     #STBOT,SP               ;GOT HERE AS A RESULT OF NXM TRAP---
 9891                                                                         ;CLEAN UP THE STACK
 9892 046170  005037  177766                  CLR     @#177766                ;CLEAR CPU ERROR REGISTER
 9893 046174  012737  046234  000004  2$:     MOV     #3$,@#4                 ;SET UP VECTOR FOR NXM TRAP
 9894 046202  052767  000020  124306          BIC     #BIT04,SR3              ;SET 18 BIT ADDRESSING MODE IN SR3
 9895 046210  012767  170000  124136          MOV     #170000,KIPAR6          ;SET KIPAR6 SO THAT BITS 18-21 SHOULD
 9896                                                                         ;BE ASSERTED IF 22 BIT ADR WAS ENABLED
 9897 046216  012737  177777  140000          MOV     #177777,@#140000        ;TRY TO WRITE ADDR 17000000 VIA KIPAR6
 9898                                                                         ;ADDR 0 SHOULD = 177777. A TIME OUT
 9899 046224  022737  177777  000000          CMP     #177777,@#0             ;OR ADDR 0 = ZERO INDICATES AN ERROR
 9900 046232  001403                          BEQ     4$                      ;GO TO NEXT TEST IF ADDR 0=177777
 9901 046234  005067  131332          3$:     CLR     SR0                     ;DISABLE MMU BEFORE ERROR.
 9902                                                                         ;ERROR! 18 BIT ADDR OPTION IS N.G.
 9903 046240  104002                          ERROR   +2              ;MMU ERROR
 9904                                 ;
 9905                                 ;TEST ADDRESS BITS 18 THRU 21
 9906                                 ;
 9907
 9908 046242  052767  000020  124246  4$:     BIS     #BIT04,SR3              ;ENABLE 22 BIT ADDRESSING MODE
 9909 046250  012767  046312  131526          MOV     #5$,4                   ;SET UP FOR NXM TRAP
 9910 046256  005067  131516                  CLR     0                       ;CLEAR ADDRESS 0
 9911 046262  012767  010000  124064          MOV     #10000,KIPAR6           ;TEST ADDRESS BIT 18
 9912 046270  012737  177777  140000          MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 1000000
 9913 046276  005767  131476                  TST     0                       ;TEST ADDRESS 0. SHOULD = ZERO
 9914 046302  001403                          BEQ     5$                      ;BRANCH IF ADDRESS 0=0
 9915 046304  005067  131262                  CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9916 046310  104002                          ERROR   +2              ;MMU ERROR
 9917                                                                 ;ERROR! BIT 18 DID NOT ASSERT
```

```
 9918 046312  012737  046354  000004  5$:     MOV     #6$,@#4                 ;SET UP FOR NXM TRAP
 9919 046320  005067  131454                  CLR     0                       ;CLEAR ADDR 0
 9920 046324  012767  020000  124022          MOV     #20000,KIPAR6           ;TEST ADDRESS BIT 19
 9921 046332  012737  177777  140000          MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 2000000
 9922 046340  005767  131434                  TST     0                       ;TEST ADDR 0. SHOULD= ZERO
 9923 046344  001403                          BEQ     6$                      ;BRANCH IF ADDRESS 0=0
 9924 046346  005067  131220                  CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9925 046352  104002                          ERROR   +2                  ;MMU ERROR
 9926                                                                      ;ERROR! BIT 19 DID NOT ASSERT
 9927 046354  012737  046416  000004  6$:     MOV     #7$,@#4                 ;SET UP FOR NXM TRAP
 9928 046362  005067  131412                  CLR     0                       ;CLEAR ADDR 0
 9929 046366  012767  040000  123760          MOV     #40000,KIPAR6           ;TEST ADDRESS BIT 20
 9930 046374  012737  177777  140000          MOV     #177777,@#140000        ;WRITE ALL ONES TO ADDR 4000000
 9931 046402  005767  131372                  TST     0                       ;TEST ADDR 0. SHOULD =0
 9932 046406  001403                          BEQ     7$                      ;BRANCH IF ADDRESS 0 =0
 9933 046410  005067  131156                  CLR     SR0                     ;DISABLE MMU BEFORE ERROR
 9934 046414  104002                          ERROR   +2                  ;MMU ERROR
 9935                                                                      ;ERROR! BIT 20 DID NOT ASSERT
 9936 046416  012737  046460  000004  7$:     MOV     #8$,@#4                 ;SET UP FOR NXM
 9937 046424  005067  131350                  CLR     0                       ;CLEAR ADDRESS 0
 9938 046430  012767  100000  123716          MOV     #100000,KIPAR6          ;TEST ADDRESS BIT 21
 9939 046436  012737  177777  140000          MOV     #177777,@#140000        ;WRITE ALL ONES AT ADDR 10000000
 9940 046444  005767  131330                  TST     0                       ;CHECK ADDRESS 0. SHOULD = 0
 9941 046450  001403                          BEQ     8$                      ;BRANCH IF ADDR 0 = 0
 9942 046452  005067  131114                  CLR     SR0                     ;DISABLE MMU    BEFORE ERROR
 9943 046456  104002                          ERROR   +2                  ;MMU ERROR
 9944                                                                      ;ERROR! ADDR BIT 21 DID NOT ASSERT
 9945 046460  005067  131106          8$:     CLR     SR0                     ;DISABLE MMU
 9946 046464  005037  177766                  CLR     @#177766                ;CLEAR CPU ERROR REGISTER
 9947 046470  012706  001000                  MOV     #STBOT,R6               ;RESET STACK POINTER
 9948 046474  013737  003012  000004          MOV     @#SLOC00,@#4            ;RESTORE VECTORS
 9949 046502  013737  003014  000006          MOV     @#SLOC01,@#6            ;
 9950
 9959 046510                                  TSMM14:
 9960                                         ;       ADDER RELOCATION TEST PART B
 9961                                         ;;**************************************************************************
                                              ;(NEED 22 BITS OF MEMORY ADDRESSING)
 9962 046510  005037  177572                  CLR     @#SR0                   ;TURN OFF MMU.
 9963 046514  005067  131246                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
 9964 046520  013737  000004  003012          MOV     @#4,@#SLOC00            ;SAVE LOC 4 IN SLOC00.
 9965 046526  013737  000006  003014          MOV     @#6,@#SLOC01            ;SAVE LOC 6 IN SLOC01.
 9966 046534  012737  047224  000004          MOV     #NXMTRP,@#4             ;SET UP FOR TIMEOUT TRAP
 9967 046542  012737  000340  000006          MOV     #340,@#6                ;SET UP FOR TIMEOUT TRAP
 9968 046550  005037  172340                  CLR     @#KIPAR0                ;SET KER PAR0 FOR 1ST 4KW OF MEMORY.
 9969 046554  012767  077406  123516          MOV     #77406,KIPDR0           ;SET KER PDR FOR 4KW R/W ACCESS.
 9970 046562  012737  177500  172354          MOV     #177500,@#KIPAR6        ;SET UP KERNEL PAGE ADDR REG 6
 9971                                                                         ;FOR HIGHEST 4K WORDS OF NON-I/O
 9972                                                                         ;FOR 2 MEG WORDS OF MEMORY.
 9973 046570  012767  077406  123516          MOV     #77406,KIPDR6           ;SET KER PDR6 FOR 4KW R/W ACCESS.
 9974 046576  012737  000020  172516          MOV     #20,@#SR3               ;ENABLE 22 BIT ADDRESSING.
 9975 046604  012737  000001  177572          MOV     #1,@#SR0                ;TURN ON THE MMU.
 9976 046612  005737  157776                  TST     @#157776                ;ATTEMPT TO ADDRESS LAST MEMORY ADDR.
 9977                                         ;**********WILL TRAP TO 4 IF 2 MEG WORDS OF MEMORY NOT AVAILABLE*********
 9978 046616  013737  003012  000004          MOV     @#SLOC00,@#4            ;RESTORE LOC 4
 9979 046624  013737  003014  000006          MOV     @#SLOC01,@#6            ;RESTORE LOC 6
 9980 046632  005037  177572                  CLR     @#177572                ;MMU OFF
 9981 046636  005037  003030                  CLR     @#FLAG                  :CLEAR MMU ABORT FLAG
```

MEMORY MANAGEMENT TESTS

```
 9982 046642  004767  066220                    JSR     PC,MMU                  ;INIT MMU
 9983 046646  012737  010000  177776            MOV     #10000,@#177776         ;INIT PSW
 9984 046654  012737  000020  172516            MOV     #20,@#172516            ;INIT MMR3
 9985 046662  052737  001000  177746            BIS     #1000,@#177746          ;TURN CACHE TEST FEATURE ON
 9986 046670  012704  047426                    MOV     #PARVA3,R4              ;SET POINTERS TO INIT TABLES
 9987 046674  012701  047460                    MOV     #VIR3,R1                ;
 9988 046700  012437  172246          1$:       MOV     (R4)+,@#172246          ;INIT SIPAR3
 9989 046704  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
 9990 046712  012746  125252                    MOV     #125252,-(SP)           ;PUSH BACKGROUND DATA ON TO THE STACK
 9991 046716  006671  000000                    MTPI    @0(R1)                  ;WRITE DATA TO PHYSICAL ADDRESS
 9992 046722  006531                            MFPI    @(R1)+                  ;WRITE DATA AT PHYSICAL ADDRESS TO STACK
 9993 046724  022726  125252                    CMP     #125252,(SP)+           ;IS DATA EQUAL TO EXPECTED
 9994 046730  001403                            BEQ     2$                      ;YES GO ON
 9995 046732  005037  177572                    CLR     @#177572                ;TURN MMU OFF
 9996 046736  104002                            ERROR   +2                      ;MMU ERROR
 9997                                                                           ;NOT EQUAL GO TO ERROR
 9998 046740  005037  177572          2$:       CLR     @#177572                ;TURN MMU OFF
 9999 046744  021427  000333                    CMP     (R4),#333               ;ARE WE DONE
10000 046750  001353                            BNE     1$                      ;NO GO TO 1$
10001 046752  012704  047310                    MOV     #PARVA2,R4              ;SET POINTERS TO PAR INIT TABLES
10002 046756  012701  047256                    MOV     #PARAD2,R1              ;
10003 046762  012431                  3$:       MOV     (R4)+,@(R1)+            ;INIT PARS
10004 046764  021127  000333                    CMP     (R1),#333               ;ARE WE DONE
10005 046770  001374                            BNE     3$                      ;NO, GO TO 3$
10006 046772  012704  047374                    MOV     #MODE2,R4               ;SET POINTERS TO ADDER PART B TABLES
10007 046776  012701  047342                    MOV     #VIR2,R1                ;
10008 047002  012702  047426                    MOV     #PARVA3,R2              ;
10009 047006  012703  047460                    MOV     #VIR3,R3                ;
10010 047012  004767  000076          4$:       JSR     PC,TS14                 ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
10011                                                                           ;CHECK IF DATA AT PHYSICAL ADDRESS IS
10012                                                                           ;EQUAL TO EXPECTED AND IF NOT DETERMINE
10013                                                                           ;IF IT IS AN ADDER ERROR OR A MEMORY ERROR
10014 047016  021127  000111                    CMP     (R1),#111               ;HAVE WE DONE ALL THE 22 BIT MODE I SPACE
10015                                                                           ;CASES
10016 047022  001373                            BNE     4$                      ;NO GO TO 4$
10017 047024  005201                            INC     R1                      ;POINT TO 22 BIT MODE D SPACE CASE
10018 047026  005201                            INC     R1                      ;
10019 047030  005204                            INC     R4                      ;
10020 047032  005204                            INC     R4                      ;
10021 047034  012737  000027  172516            MOV     #27,@#172516            ;INIT MMR3
10022 047042  012437  177776                    MOV     (R4)+,@#177776          ;INIT PSW
10023 047046  012746  052525                    MOV     #52525,-(SP)            ;PUSH DATA ONTO STACK
10024 047052  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
10025 047060  106631                            MTPD    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS
10026 047062  005037  177572                    CLR     @#177572                ;TURN MMU OFF
10027 047066  012737  000020  172516            MOV     #20,@#172516            ;INIT MMR3
10028 047074  004767  000040                    JSR     PC,T14                  ;CHECK IF DATA AT PHYSICAL ADDRESS IS EQUAL
10029                                                                           ;TO EXPECTED AND IF NOT DETERMINE IF IT
10030                                                                           ;IS AN ADDER ERROR OR A MEMORY ERROR
10031 047100  005037  172516                    CLR     @#172516                ;INIT MMR3 FOR 18 BIT MODE
10032 047104  004767  000004                    JSR     PC,TS14                 ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
10033                                                                           ;CHECK IF DATA AT PHYSICAL ADDRESS IS
10034                                                                           ;EQUAL TO EXPECTED AND IF NOT DETERMINE IF
10035                                                                           ;IT IS AN ADDER ERROR OR A MEMORY ERROR
10036
10037 047110  000167  000376                    JMP     T14FIN
10038                                 ;
```

MEMORY MANAGEMENT TESTS

```
10039                                   ;ROUTINE TO WRITE DATA TO PHYSICAL ADDRESS AND TO CHECK IF DATA AT
10040                                   ;PHYSICAL ADDRESS IS EQUAL TO EXPECTED AND IF NOT DETERMINE IF IT IS
10041                                   ;AN ADDER ERROR OR A MEMORY ERROR
10042                                   ;
10043 047114  012437  177776      TS14:   MOV     (R4)+,@#177776          ;INIT PSW
10044 047120  012737  000001  177572      MOV     #1,@#177572             ;TURN MMU ON
10045 047126  012746  052525              MOV     #52525,-(SP)            ;WRITE DATA ONTO STACK
10046 047132  006631                      MTPI    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS VIA STACK
10047 047134  005037  177572              CLR     @#177572                ;TURN MMU OFF
10048 047140  012737  010000  177776  T14:    MOV     #10000,@#177776         ;INIT PSW
10049 047146  012237  172246              MOV     (R2)+,@#172246          ;INIT SIPAR3
10050 047152  012737  000001  177572      MOV     #1,@#177572             ;TURN MMU ON
10051 047160  006573  000000              MFPI    @0(R3)                  ;DO RELOCATION
10052 047164  022726  052525              CMP     #52525,(SP)+            ;IS DATA EQUAL TO EXPECTED
10053 047170  001410                      BEQ     2$                      ;YES GO ON
10054 047172  006573  000000              MFPI    @0(R3)                  ;WHAT TYPE OF ERROR IS IT
10055 047176  022726  125252              CMP     #125252,(SP)+           ;
10056 047202  001402                      BEQ     1$                      ;
10057 047204  104002                      ERROR   +2              ;MMU ERROR
10058                                                           ;IT IS A MEMORY ERROR
10059 047206  000401                      BR      2$                      ;
10060 047210  104002              1$:     ERROR   +2              ;MMU ERROR
10061                                                           ;IT IS AN ADDER ERROR
10062 047212  005037  177572      2$:     CLR     @#177572                ;TURN MMU OFF
10063 047216  005203                      INC     R3                      ;
10064 047220  005203                      INC     R3                      ;
10065 047222  000207                      RTS     PC                      ;RETURN
10066                                   ;
10067                                   ;NON-EXISTANT MEMORY TRAP ROUTINE
10068                                   ;
10069 047224  005037  177572      NXMTRP: CLR     @#SR0                   ;TURN OFF MMU.
10070 047230  012716  047512              MOV     #T14FIN,(SP)            ;SET UP STACK WITH RETURN ADDR.
10071 047234  013737  003012  000004      MOV     @#SLOC00,@#4            ;RESTORE LOC 4
10072 047242  013737  003014  000006      MOV     @#SLOC01,@#6            ;RESTORE LOC 6
10073 047250  005037  177766              CLR     @#177766                ;CLEAR TIME OUT INDICATION FROM
10074                                                                   ;CPU ERROR REGISTER.
10075 047254  000006                      RTT                             ;RETURN FROM TRAP; GO TO NEXT TEST.
10076                                   ;
10077                                   ;ADDER TEST PART B TABLES
10078                                   ;
10079 047256  177646      PARAD2: .WORD   177646
10080 047260  177650              .WORD   177650
10081 047262  177652              .WORD   177652
10082 047264  172240              .WORD   172240
10083 047266  177640              .WORD   177640
10084 047270  177642              .WORD   177642
10085 047272  172244              .WORD   172244
10086 047274  177644              .WORD   177644
10087 047276  172252              .WORD   172252
10088 047300  172352              .WORD   172352
10089 047302  177662              .WORD   177662
10090 047304  172242              .WORD   172242
10091 047306  000333              .WORD   333
10092 047310  157700      PARVA2: .WORD   157700
10093 047312  137700              .WORD   137700
10094 047314  077700              .WORD   077700
10095 047316  176777              .WORD   176777
```

MEMORY MANAGEMENT TESTS

```
10096 047320  007600                          .WORD   007600
10097 047322  167700                          .WORD   167700
10098 047324  175700                          .WORD   175700
10099 047326  177425                          .WORD   177425
10100 047330  177220                          .WORD   177220
10101 047332  173700                          .WORD   173700
10102 047334  176700                          .WORD   176700
10103 047336  077400                          .WORD   077400
10104 047340  000333                          .WORD   333
10105 047342  070000                  VIR2:   .WORD   070000
10106 047344  110000                          .WORD   110000
10107 047346  130000                          .WORD   130000
10108 047350  000000                          .WORD   000000
10109 047352  000000                          .WORD   000000
10110 047354  030000                          .WORD   030000
10111 047356  050000                          .WORD   050000
10112 047360  052524                          .WORD   052524
10113 047362  136000                          .WORD   136000
10114 047364  130000                          .WORD   130000
10115 047366  000111                          .WORD   111
10116 047370  030000                          .WORD   030000
10117 047372  030000                          .WORD   030000
10118 047374  030000                  MODE2:  .WORD   030000
10119 047376  030000                          .WORD   030000
10120 047400  030000                          .WORD   030000
10121 047402  010000                          .WORD   010000
10122 047404  030000                          .WORD   030000
10123 047406  030000                          .WORD   030000
10124 047410  010000                          .WORD   010000
10125 047412  030000                          .WORD   030000
10126 047414  010000                          .WORD   010000
10127 047416  000000                          .WORD   000000
10128 047420  000111                          .WORD   111
10129 047422  030000                          .WORD   030000
10130 047424  010000                          .WORD   010000
10131 047426  160000                  PARVA3: .WORD   160000
10132 047430  140000                          .WORD   140000
10133 047432  100000                          .WORD   100000
10134 047434  176770                          .WORD   176770
10135 047436  007600                          .WORD   007600
10136 047440  170000                          .WORD   170000
10137 047442  176000                          .WORD   176000
10138 047444  177552                          .WORD   177552
10139 047446  177400                          .WORD   177400
10140 047450  174000                          .WORD   174000
10141 047452  177000                          .WORD   177000
10142 047454  007500                          .WORD   007500
10143 047456  000333                          .WORD   333
10144 047460  060000                  VIR3:   .WORD   060000
10145 047462  060000                          .WORD   060000
10146 047464  060000                          .WORD   060000
10147 047466  060700                          .WORD   060700
10148 047470  060000                          .WORD   060000
10149 047472  060000                          .WORD   060000
10150 047474  060000                          .WORD   060000
10151 047476  060024                          .WORD   060024
10152 047500  060000                          .WORD   060000
```

MEMORY MANAGEMENT TESTS

```
10153 047502  060000                          .WORD   060000
10154 047504  060000                          .WORD   060000
10155 047506  060000                          .WORD   060000
10156 047510  000333                          .WORD   333
10157
10158
10159                                 ;
10160 047512                          T14FIN:
10173
10174                                 ;       TEST NON-EXISTANT MEMORY TRAP
10175                                 ;;**********************************************************************
                                      ;WE ARE ASSUMING THAT THE NON-EXISTANT MEMORY TIME OUT
                                      ;FEATURE IS WORKING SINCE WE CAN'T GUARANTEE THAT
                                      ;THE SYSTEM BEING TESTED HAS A NON-EXISTANT MEMORY LOCATION.
                                      ;AT THIS TIME WE WILL ATTEMPT TO TEST THE NXM FUNCTION
10176 047512  004767  065350                  JSR     PC,MMU                  ;INIT THE MMU
10177 047516  012737  177400  172354          MOV     #177400,@#KIPAR6        ;SET KIPAR6 TO RELOCATE TO HIGHEST MEMORY
10178 047524  016767  130254  133260          MOV     4,SLOC00                ;SAVE VECTOR
10179 047532  016767  000026  130244          MOV     2$,4                    ;LOAD VEC WITH ADDR OF TRAP HANDLER
10180 047540  052767  000001  130024          BIS     #BIT00,SR0              ;TURN ON THE MMU
10181 047546  005067  130214                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10182 047552  005067  130220                  CLR     PS                      ;CLEAR THE PSW
10183 047556  005737  157776                  TST     @#157776                ;ACCESS PHYSICAL ADDR 17757776
10184 047562  000415                  1$:     BR      NXMFIN                  ;IF IT DOESN'T TRAP WE'LL ASSUME
10185                                                                         ;THAT THIS IA A 4 MEGABYTE SYSTEM
10186                                                                         ;AND GO TO THE NEXT TEST
10187 047564  022767  000040  130174  2$:     CMP     #BIT05,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
10188 047572  001401                          BEQ     3$                      ;
10189 047574  104002                          ERROR   +2              ;MMU ERROR
10190 047576  022726  047562          3$:     CMP     #1$,(SP)+               ;IS CONTENTS OF STACK CORRECT?
10191 047602  001401                          BEQ     4$                      ;
10192 047604  104002                          ERROR   +2              ;MMU ERROR
10193 047606  022726  000000          4$:     CMP     #0,(SP)+                ;IS CONTENTS OF STACK CORRECT?
10194 047612  001401                          BEQ     NXMFIN                  ;
10195 047614  104002                          ERROR   +2              ;MMU ERROR
10196 047616  005067  127750          NXMFIN: CLR     SR0                     ;TURN OFF THE MMU
10197 047622  005067  130140                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10198 047626  016767  133160  130150          MOV     SLOC00,4                ;RESTORE THE VECTOR
10199
10201
10203 047634                          TSMM15:
10204                                 ;       PAGE WRITTEN BIT TEST
10205 047634  005037  177572                  CLR     @#177572                ;MMU OFF
10206 047640  005067  133164                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
10207 047644  004767  065216                  JSR     PC,MMU                  ;INIT MMU
10208 047650  005037  177776                  CLR     @#177776                ;INIT PSW
10209 047654  012704  172300                  MOV     #172300,R4              ;SET POINTER TO KPDRS
10210 047660  004767  000114                  JSR     PC,TS15                 ;DO RELOCATIONS AND TEST KIPDRS FOR
10211                                                                         ;PAGE WRITTEN BIT BEING SET AND IF
10212                                                                         ;NOT SET GO TO ERROR
10213 047664  004767  000160                  JSR     PC,T15                  ;DO RELOCATIONS AND TEST KDPDRS FOR
10214                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10215                                                                         ;SET GO TO ERROR
10216 047670  012737  050000  177776          MOV     #50000,@#177776         ;INIT PSW
10217 047676  012704  172200                  MOV     #172200,R4              ;SET POINTER TO SPDRS
10218 047702  004767  000072                  JSR     PC,TS15                 ;DO RELOCATIONS AND TEST SIPDRS FOR
10219                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
```

MEMORY MANAGEMENT TESTS

```
10220
10221 047706  004767  000136                JSR     PC,T15                  ;SET GO TO ERROR
10222                                                                         ;DO RELOCATIONS AND TEST SDPDRS FOR
10223                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10224 047712  005037  177776                CLR     @#177776                ;SET GO TO ERROR
10225 047716  012737  170000  177776        MOV     #170000,@#177776        ;INIT PSW TO A KNOWN STATE
10226 047724  012704  177600                MOV     #177600,R4              ;INIT PSW
10227 047730  004767  000044                JSR     PC,TS15                 ;SET POINTER TO UPDRS
10228                                                                         ;DO RELOCATIONS AND TEST UIPDRS FOR
10229                                                                         ;PAGE WRITTEN BIT BEING SET AND IF
10230 047734  004767  000110                JSR     PC,T15                  ;NOT SET GO TO ERROR
10231                                                                         ;DO RELOCATIONS AND TEST UDPDRS FOR
10232                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10233 047740  005037  177776                CLR     @#177776                ;SET GO TO ERROR
10234 047744  012704  172300                MOV     #172300,R4              ;INIT PSW TO A KNOWN STATE
10235 047750  004767  000152                JSR     PC,T15A                 ;SET POINTER TO KPDRS
10236                                                                         ;EXPLICITLY WRITE TO KPDRS AND TEST
10237                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10238 047754  012704  172200                MOV     #172200,R4              ;AND IF NOT CLEARED GO TO ERROR
10239 047760  004767  000142                JSR     PC,T15A                 ;SET POINTER TO SPDRS
10240                                                                         ;EXPLICITLY WRITE TO SPDRS AND TEST
10241                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10242 047764  012704  177600                MOV     #177600,R4              ;AND IF NOT CLEARED GO TO ERROR
10243 047770  004767  000132                JSR     PC,T15A                 ;SET POINTER TO UPDRS
10244                                                                         ;EXPLICITLY WRITE TO UPDRS AND TEST
10245                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10246                                                                         ;AND IF NOT CLEARED GO TO ERROR
10247 047774  000167  000154                JMP     T15FIN
10248                                 ;
10249                                 ;ROUTINE TO DO RELOCATIONS AND TEST IPDRS FOR PAGE WRITTEN BIT BEING
10250                                 ;SET AND IF NOT SET REPORT AN ERROR
10251                                 ;
10252 050000  005001                TS15:   CLR     R1                      ;SET POINTER TO VIRTUAL ADDRESS
10253 050002  012737  000020  172516        MOV     #20,@#172516            ;INIT MMR3
10254 050010  012737  000001  177572  1$:   MOV     #1,@#177572             ;TURN MMU ON
10255 050016  011111                        MOV     (R1),(R1)               ;DO A RELOCATION
10256 050020  005037  177572                CLR     @#177572                ;TURN MMU OFF
10257 050024  022427  077506                CMP     (R4)+,#77506            ;IS DATA EQUAL TO EXPECTED
10258 050030  001401                        BEQ     2$                      ;OK GO ON
10259 050032  104002                        ERROR   .+2                 ;MMU ERROR
10260                                                                       ;NO GO TO ERROR
10261 050034  062701  020000          2$:   ADD     #20000,R1               ;POINT TO NEXT VIRTUAL ADDRESS
10262 050040  020127  160000                CMP     R1,#160000              ;ARE WE DONE
10263 050044  001361                        BNE     1$                      ;NO GO TO 1$
10264 050046  000207                        RTS     PC                      ;RETURN
10265                                 ;
10266                                 ;ROUTINE TO DO RELOCATIONS AND TEST DPDRS FOR PAGE WRITTEN BIT BEING SET
10267                                 ;AND IF NOT SET REPORT AN ERROR
10268                                 ;
10269 050050  005001                T15:    CLR     R1                      ;SET POINTER TO VIRTUAL ADDRESS
10270 050052  062704  000002                ADD     #2,R4                   ;POINT TO FIRST DPDR
10271 050056  012737  000027  172516        MOV     #27,@#172516            ;INIT MMR3
10272 050064  012737  000001  177572  1$:   MOV     #1,@#177572             ;TURN MMU ON
10273 050072  011146                        MOV     (R1),-(SP)              ;PUSH DATA ONTO THE STACK
10274 050074  106611                        MTPD    (R1)                    ;DO A RELOCATION
10275 050076  005037  177572                CLR     @#177572                ;TURN MMU OFF
10276 050102  022427  077506                CMP     (R4)+,#77506            ;IS DATA EQUAL TO EXPECTED
```

MEMORY MANAGEMENT TESTS

```
10277 050106  C01401                        BEQ     2$                      ;OK GO ON
10278 050110  104002                        ERROR   .2                  ;MMU ERROR
10279                                                                    ;NO GO TO ERROR
10280 050112  062701  020000          2$:   ADD     #20000,R1               ;POINT TO NEXT VIRTUAL ADDRESS
10281 050116  020127  160000                CMP     R1,#160000              ;ARE WE DONE
10282 050122  001360                        BNE     1$                      ;NO GO TO 1$
10283 050124  000207                        RTS     PC                      ;RETURN
10284                                 ;
10285                                 ;ROUTINE TO EXPLICITLY WRITE TO PDRS AND TEST PAGE WRITTEN BIT FOR BEING
10286                                 ;CLEARED AND IF NOT CLEARED REPORT AN ERROR
10287                                 ;
10288 050126  005002                  T15A: CLR     R2                      ;CLEAR COUNTER
10289 050130  011414                  1$:   MOV     (R4),(R4)               ;DO AN EXPLICIT WRITE TO PDR
10290 050132  022427  077406                CMP     (R4)+,#77406            ;IS DATA EQUAL TO EXPECTED
10291 050136  001401                        BEQ     2$                      ;OK GO ON
10292 050140  104002                        ERROR   .2                  ;MMU ERROR
10293                                                                    ;NO GO TO ERROR
10294 050142  005202                  2$:   INC     R2                      ;INCREMENT POINTER
10295 050144  020227  000020                CMP     R2,#20                  ;ARE WE DONE
10296 050150  001367                        BNE     1$                      ;NO GO TO 1$
10297 050152  000207                        RTS     PC                      ;RETURN
10298 050154                          T15FIN:
10299                                 ;
10302 050154                          TSMM16:
10303                                 ;       TEST CSM (CALL SUPERVISOR MODE)
10304 050154  005037  177572                CLR     @#177572                ;MMU OFF
10305 050160  005037  003030                CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
10306 050164  012704  050504                MOV     #TMM16E,R4              ;INIT R4
10307 050170  004767  064672                JSR     PC,MMU                  ;INIT MMU
10308 050174  012737  000037  172516        MOV     #37,@#172516            ;ENABLE CSM INSTRUCTION
10309 050202  005037  177776                CLR     @#177776                ;SET PS TO KER MODE
10310 050206  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTORS
10311 050212  013746  000014                MOV     @#14,-(SP)              ;
10312 050216  013746  000016                MOV     @#16,-(SP)              ;
10313 050222  012737  050374  000010        MOV     #TMM16B,@#10            ;SETUP NEW VECTORS
10314 050230  012737  000137  000014        MOV     #137,@#14               ;
10315 050236  012737  050514  000016        MOV     #TMM16A,@#16            ;
10316 050244  007014                        .WORD   7014                    ; TEST INSTRUCTION
10317 050246  104002                        ERROR   .2                  ;MMU ERROR
10318                                                                    ;GO TO ERROR IF NOT TRAPPED
10319 050250  012737  050424  000010  TSM16A: MOV     #TMM16C,@#10            ;SETUP NEW VECTOR
10320 050256  012737  000027  172516        MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
10321 050264  012737  140000  177776        MOV     #140000,@#177776        ;SET PS TO USER MODE
10322 050272  007014                        .WORD   7014                    ; TEST INSYRUCTION
10323 050274  104002                        ERROR   .2                  ;MMU ERROR
10324                                                                    ;GO TO ERROR IF NOT TRAPPED
10325 050276  012737  050454  000010  TSM16B: MOV     #TMM16D,@#10            ;SETUP NEW VECTOR
10326 050304  012737  000027  172516        MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
10327 050312  005037  177776                CLR     @#177776                ;SET PS TO KER MODE
10328 050316  007014                        .WORD   7014                    ; TEST INSTRUCTION
10329 050320  104002                        ERROR   .2                  ;MMU ERROR
10330                                                                    ;GO TO ERROR IF NOT TRAPPED
10331 050322  012737  000037  172516  TSM16C: MOV     #37,@#172516            ;ENABLE CSM INSTRUCTION
10332 050330  012737  040000  177776        MOV     #40000,@#177776         ;SET PS TO SUP MODE
10333 050336  012706  000700                MOV     #700,R6                 ;INIT SUP SP
10334 050342  012737  140000  177776        MOV     #140000,@#177776        ;SET PS TO USER MODE
10335 050350  012706  000600                MOV     #600,R6                 ;INIT USER SP
```

MEMORY MANAGEMENT TESTS

```
10336 050354  012737  000014  000010          MOV     #14,@#10                ;SETUP NEW VECTOR
10337 050362  000277                          SCC                             ;SET ALL CC BITS
10338 050364  007024                          .WORD   7024                    ; TEST INSTRUCTION
10339 050366  104002                  TSM16D: ERROR   .2              ;MMU ERROR
10340                                                                 ;GO TO ERROR IF NOT TRAPPED
10341 050370  000167  000504                  JMP     TM16A
10342                                 ;
10343                                 ;
10344 050374  042737  007777  177776  TMM16B: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10345 050402  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10346 050410  001401                          BEQ     1$                      ;YES GO ON
10347 050412  104002                          ERROR   .2              ;MMU ERROR
10348                                                                 ;NO GO TO ERROR
10349 050414  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10350 050416  005726                          TST     (SP)+                   ;
10351 050420  000167  177624                  JMP     TSM16A                  ;CONTINUE TESTING
10352 050424  042737  007777  177776  TMM16C: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10353 050432  022737  030000  177776          CMP     #30000,@#177776         ;IS PS CORRECT
10354 050440  001401                          BEQ     1$                      ;YES GO ON
10355 050442  104002                          ERROR   .2              ;MMU ERROR
10356                                                                 ;NO GO TO ERROR
10357 050444  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10358 050446  005726                          TST     (SP)+                   ;
10359 050450  000167  177622                  JMP     TSM16B                  ;CONTINUE TESTING
10360 050454  042737  007777  177776  TMM16D: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10361 050462  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10362 050470  001401                          BEQ     1$                      ;YES GO ON
10363 050472  104002                          ERROR   .2              ;MMU ERROR
10364                                                                 ;NO GO TO ERROR
10365 050474  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10366 050476  005726                          TST     (SP)+                   ;
10367 050500  000167  177616                  JMP     TSM16C                  ;CONTINUE TESTING
10368 050504  156430                  TMM16E: .WORD   156430                  ; TEST LOCATION
10369 050506  104002                  TMM16F: ERROR   .2              ;MMU ERROR
10370                                                                 ;GO TO ERROR IF DIDN'T ABORT
10371 050510  000167  000364                  JMP     TM16A
10372 050514  022737  070017  177776  TMM16A: CMP     #70017,@#177776         ;IS PS CORRECT
10373 050522  001401                          BEQ     1$                      ;YES GO ON
10374 050524  104002                          ERROR   .2              ;MMU ERROR
10375                                                                 ;NO GO TO ERROR
10376 050526  020627  000572          1$:     CMP     R6,#572                 ;IS SP CORRECT
10377 050532  001401                          BEQ     2$                      ;YES GO ON
10378 050534  104002                          ERROR   .2              ;MMU ERROR
10379                                                                 ;NO GO TO ERROR
10380 050536  020427  050506          2$:     CMP     R4,#TMM16E+2            ;IS R4 CORRECT
10381 050542  001401                          BEQ     3$                      ;YES GO ON
10382 050544  104002                          ERROR   .2              ;MMU ERROR
10383                                                                 ;NO GO TO ERROR
10384 050546  023727  050504  156430  3$:     CMP     @#TMM16E,#156430        ;IS TEST LOCATION OK
10385 050554  001401                          BEQ     4$                      ;YES GO ON
10386 050556  104002                          ERROR   .2              ;MMU ERROR
10387                                                                 ;NO GO TO ERROR
10388 050560  022627  156430          4$:     CMP     (SP)+,#156430           ;IS STACK CORRECT
10389 050564  001401                          BEQ     5$                      ;YES GO ON
10390 050566  104002                          ERROR   .2              ;MMU ERROR
10391                                                                 ;NO GO TO ERROR
10392 050570  022627  050366          5$:     CMP     (SP)+,#TSM16D           ;IS STACK CORRECT
```

```
10393 050574  001401                        BEQ     6$                      ;YES GO ON
10394 050576  104002                        ERROR   .+2                  ;MMU ERROR
10395                                                                     ;NO GO TO ERROR
10396 050600  022627  140000          6$:   CMP     (SP)+,#140000           ;IS STACK CORRECT
10397 050604  001401                        BEQ     7$                      ;YES GO ON
10398 050606  104002                        ERROR   .+2                  ;MMU ERROR
10399                                                                     ;NO GO TO ERROR
10400 050610  012706  000700          7$:   MOV     #700,R6                 ;RESTORE SUP SP
10401 050614  012737  140000  177776        MOV     #140000,@#177776        ;SET PS TO USER MODE
10402 050622  020627  000600                CMP     R6,#600                 ;IS USER SP CORRECT
10403 050626  001401                        BEQ     8$                      ;YES GO ON
10404 050630  104002                        ERROR   .+2                  ;MMU ERROR
10405                                                                     ;NO GO TO ERROR
10406 050632  012767  077400  121340  8$:   MOV     #77400,SIPDR0           ;SETUP SIPDR0 TO ABORT
10407 050640  012737  050506  000016        MOV     #TMM16F,@#16            ;SETUP VECTOR
10408 050646  012737  000001  003030        MOV     #1,@#FLAG               ;SETUP FLAG FOR AN ABORT
10409 050654  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
10410 050662  010701                        MOV     R7,R1                   ;SAVE OLD PC
10411 050664  007014                        .WORD   7014                    ; TEST INSTRUCTION
10412 050666  022737  000000  003030        CMP     #0,@#FLAG               ;DID AN ABORT OCCUR
10413 050674  001401                        BEQ     9$                      ;YES GO ON
10414 050676  104002                        ERROR   .+2                  ;MMU ERROR
10415                                                                     ;NO GO TO ERROR
10416 050700  023701  003046          9$:   CMP     @#SAVMR2,R1             ;IS MMR2 CORRECT
10417 050704  001401                        BEQ     10$                     ;YES GO ON
10418 050706  104002                        ERROR   .+2                  ;MMU ERROR
10419                                                                     ;NO GO TO ERROR
10420 050710  023727  003042  100041  10$:   CMP     @#SAVMR0,#100041        ;IS MMR0 CORRECT
10421 050716  001401                        BEQ     11$                     ;YES GO ON
10422 050720  104002                        ERROR   .+2                  ;MMU ERROR
10423                                                                     ;NO GO TO ERROR
10424 050722  012737  000037  172516  11$:   MOV     #37,@#172516            ;ENABLE CSM
10425 050730  012737  040000  177776        MOV     #40000,@#177776         ;SET PSW TO SUP
10426 050736  012706  000700                MOV     #700,R6                 ;SETUP SUP SP
10427 050742  012737  140000  177776        MOV     #140000,@#177776        ;SET PSW TO USE
10428 050750  012706  000600                MOV     #600,R6                 ;SETUP USE SP
10429 050754  012737  000014  000010        MOV     #14,@#10                ;SETUP NEW VECTOR
10430 050762  012737  051004  000016        MOV     #TS16,@#16              ;SETUP NEW VECTOR
10431 050770  000277                        SCC                             ;SET ALL CC BITS
10432 050772  007027                        .WORD   7027                    ;TEST INSTRUCTION
10433 050774  045712                        .WORD   45712
10434 050776  104002          TS16A:  ERROR   .+2                  ;MMU ERROR
10435                                                                     ;GO TO ERROR IF DIDN'T TRAP
10436 051000  000167  000074                JMP     TM16A
10437 051004  022737  070017  177776  TS16:  CMP     #70017,@#177776         ;IS PSW CORRECT
10438 051012  001401                        BEQ     200$                    ;YES GO ON
10439 051014  104002                        ERROR   .+2                  ;MMU ERROR
10440                                                                     ;NO GO TO ERROR
10441 051016  020627  000572          200$:  CMP     R6,#572                 ;IS SP CORRECT
10442 051022  001401                        BEQ     201$                    ;YES GO ON
10443 051024  104002                        ERROR   .+2                  ;MMU ERROR
10444                                                                     ;NO GO TO ERROR
10445 051026  022627  045712          201$:  CMP     (SP)+,#45712            ;IS STACK CORRECT
10446 051032  001401                        BEQ     202$                    ;YES GO ON
10447 051034  104002                        ERROR   .+2                  ;MMU ERROR
10448                                                                     ;NO GO TO ERROR
10449 051036  022627  050776          202$:  CMP     (SP)+,#TS16A            ;IS STACK CORRECT
```

MEMORY MANAGEMENT TESTS

```
10450 051042  001401                        BEQ     203$                    ;YES GO ON
10451 051044  104002                        ERROR   .2                ;MMU ERROR
10452                                                                 ;NO GO TO ERROR
10453 051046  022627  140000          203$:   CMP     (SP)+,#140000           ;IS STACK CORRECT
10454 051052  001401                        BEQ     204$                    ;YES GO ON
10455 051054  104002                        ERROR   .2                ;MMU ERROR
10456                                                                 ;NO GO TO ERROR
10457 051056  012706  000700          204$:   MOV     #700,R6                 ;RESTORE SUP SP
10458 051062  012737  140000  177776        MOV     #140000,@#177776        ;SET PSW TO USER MODE
10459 051070  020627  000600                CMP     R6,#600                 ;IS USER SP CORRECT
10460 051074  001401                        BEQ     TM16A                   ;YES GO ON
10461 051076  104002                        ERROR   .2                ;MMU ERROR
10462                                                                 ;NO GO TO ERROR
10463 051100  005037  177776          TM16A:  CLR     @#177776                ;SET PS TO KER MODE
10464 051104  012637  000016                MOV     (SP)+,@#16              ;RESTORE VECTORS
10465 051110  012637  000014                MOV     (SP)+,@#14              ;
10466 051114  012637  000010                MOV     (SP)+,@#10              ;
10467
10471                                       ;*****************************************************************************
10472                                       .SBTTL  FLOATING POINT TESTS
10473                                       ;*****************************************************************************
10474                                       ;                       BEGIN FLOATING POINT TESTING
10475                                       ;*****************************************************************************
10476                                       ;*****************************************************************************
10489 051120                                MBT1:
10490
10491                                       ;       FPP REGISTER BIT TESTS
10492                                       ;;*****************************************************************
                                            ;R5=FPP POINTER
                                            ;R1=TEMPORARY COUNTER
                                            ;R2=POINTER TO EXPECTED DATA
                                            ;R3=POINTER TO RECEIVED DATA
                                            ;R4=ODD/EVEN COUNTER
10493 051120  170011                        SETD
10494 051122  005005                MBT2:   CLR     R5                      ;SETUP FPP ACC POINTER
10495 051124  012702  003100                MOV     #BTEXP,R2               ;POINT TO TEST DATA
10496 051130  012703  003110                MOV     #BTRES,R3               ;POINT TO RECEIVED DATA
10497 051134  170400                MBT2A:  CLRD    AC0                     ;SETUP FPP REGISTER VALUES
10498 051136  174012                        STD     AC0,(R2)                ;CLEAR EXPECTED VALUE
10499 051140  005004                        CLR     R4
10500 051142  170400                BTG0:   CLRD    AC0                     ;SETUP FPP REGISTER VALUES
10501 051144  170401                        CLRD    AC1
10502 051146  170402                        CLRD    AC2
10503 051150  170403                        CLRD    AC3
10504 051152  170404                        CLRD    AC4
10505 051154  170405                        CLRD    AC5
10506
10507 051156  010501                        MOV     R5,R1                   ;GET FPP AC NUMBER INTO R1
10508 051160  070127  000014                MUL     #14,R1                  ;ALLOW 10 LOCATIONS FOR OPERATION
10509 051164  062701  051172                ADD     #MAC0,R1                ;SETUP JMP LOCATION
10510 051170  000111                        JMP     (R1)
10511 051172  172467  131702          MAC0:   LDD     BTEXP,AC0               ;LOAD TEST DATA INTO TEST REGISTER
10512 051176  174067  131706          MAC0A:  STD     AC0,BTRES               ;SAVE TEST RESULT
10513 051202  000167  000074                JMP     MACE                    ;GET OUT
10514 051206  172567  131666          MAC1:   LDD     BTEXP,AC1               ;LOAD TEST DATA INTO TEST REGISTER
10515 051212  174167  131672                STD     AC1,BTRES               ;SAVE TEST RESULT
10516 051216  000167  000060                JMP     MACE                    ;GET OUT
```

FLOATING POINT TESTS

```
10517 051222  172667  131652          MAC2:   LDD     BTEXP,AC2               ;LOAD TEST DATA INTO TEST REGISTER
10518 051226  174267  131656                  STD     AC2,BTRES               ;SAVE TEST RESULT
10519 051232  000167  000044                  JMP     MACE                    ;GET OUT
10520 051236  172767  131636          MAC3:   LDD     BTEXP,AC3               ;LOAD TEST DATA INTO TEST REGISTER
10521 051242  174367  131642                  STD     AC3,BTRES               ;SAVE TEST RESULT
10522 051246  000167  000030                  JMP     MACE                    ;GET OUT
10523 051252  172467  131622          MAC4:   LDD     BTEXP,AC0
10524 051256  174004                          STD     AC0,AC4                 ;LOAD TEST DATA INTO TEST REGISTER
10525 051260  172404                          LDD     AC4,AC0                 ;SAVE TEST RESULT
10526 051262  000167  177710                  JMP     MAC0A                   ;GET OUT
10527 051266  172467  131606          MAC5:   LDD     BTEXP,AC0               ;LOAD TEST DATA INTO TEST XFER REGISTER
10528 051272  174005                          STD     AC0,AC5                 ;LOAD TEST REGISTER
10529 051274  172405                          LDD     AC5,AC0                 ;STORE RESULT INTO XFER FPP REGISTER
10530 051276  000167  177674                  JMP     MAC0A                   ;GET OUT
10531 051302  026767  131572  131600  MACE:   CMP     BTEXP,BTRES
10532 051310  001014                          BNE     BTER                    ;BRANCH IF REGISTER ERROR
10533 051312  026767  131564  131572          CMP     BTEXP+2,BTRES+2
10534 051320  001010                          BNE     BTER
10535 051322  026767  131556  131564          CMP     BTEXP+4,BTRES+4
10536 051330  001004                          BNE     BTER
10537 051332  026767  131550  131556          CMP     BTEXP+6,BTRES+6
10538 051340  001401                          BEQ     MBT8                    ;GOOD RESULT
10539 051342  104003                  BTER:   ERROR   +3              ;FPP ERROR
10540                                                                 ;FPP AC LOADED INCORRECTLY
10541                                         ;NOW VERIFY THE OTHER REGISTERS REMAINED ZERO
10542 051344                                  MBT8:
10543 051344  005001                          CLR     R1                      ;CLEAR TEMPORARY COUNTER
10544 051346  005705                          TST     R5                      ;SEE IF R0 UNDER TEST
10545 051350  001411                          BEQ     MBT8A                   ;BRANCH IF TEST ING R0
10546 051352  020527  000004                  CMP     R5,#4                   ;SEE IF TESTING FPP REGISTER >R4
10547 051356  100006                          BPL     MBT8A                   ;SKIP R0 TESTING
10548 051360  174067  131524                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10549 051364  004767  000216                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10550 051370  001401                          BEQ     MBT8A                   ;BRANCH IF EXPECTED RESULT
10551 051372  104003                          ERROR   +3              ;FPP ERROR
10552                                                                 ;BAD AC0
10553 051374  020527  000001          MBT8A:  CMP     R5,#1                   ;SEE IF R1 UNDER TEST
10554 051400  001406                          BEQ     MBT8B                   ;BRANCH IF R1 UNDER TEST
10555 051402  174167  131502                  STD     AC1,BTRES               ;SAVE AC TEST RESULT
10556 051406  004767  000174                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10557 051412  001401                          BEQ     MBT8B                   ;BRANCH IF GOOD
10558 051414  104003                          ERROR   +3              ;FPP ERROR
10559                                                                 ;BAD AC1
10560 051416  020527  000002          MBT8B:  CMP     R5,#2                   ;SEE IF TESTING FPP REGISTER AC2
10561 051422  001406                          BEQ     MBT8C                   ;BRANCH IF R2 UNDER TEST
10562 051424  174267  131460                  STD     AC2,BTRES               ;SAVE AC TEST RESULT
10563 051430  004767  000152                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10564 051434  001401                          BEQ     MBT8C                   ;BRANCH IF GOOD
10565 051436  104003                          ERROR   +3              ;FPP ERROR
10566                                                                 ;BAD AC2
10567 051440  020527  000003          MBT8C:  CMP     R5,#3                   ;SEE IF R3 UNDER TEST
10568 051444  001406                          BEQ     MBT8D                   ;BRANCH IF R3 UNDER TEST
10569 051446  174367  131436                  STD     AC3,BTRES               ;SAVE AC TEST RESULT
10570 051452  004767  000130                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10571 051456  001401                          BEQ     MBT8D                   ;BRANCH IF GOOD
10572 051460  104003                          ERROR   +3              ;FPP ERROR
10573                                                                 ;BAD AC3
```

FLOATING POINT TESTS

```
10574 051462  020527  000004          MBT8D:  CMP     R5,#4                   ;SEE IF R4 UNDER TEST
10575 051466  001407                          BEQ     MBT8E                   ;BRANCH IF R4 UNDER TEST
10576 051470  172404                          LDD     AC4,AC0                 ;MOVE REGISTER CONTENT
10577 051472  174067  131412                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10578 051476  004767  000104                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10579 051502  001401                          BEQ     MBT8E                   ;BRANCH IF GOOD
10580 051504  104003                          ERROR   .3                      ;FPP ERROR
10581                                                                         ;BAD AC4
10582 051506  020527  000005          MBT8E:  CMP     R5,#5                   ;SEE IF R0 UNDER TEST
10583 051512  001407                          BEQ     MBT8F                   ;BRANCH IF R0 UNDER TEST
10584 051514  172405                          LDD     AC5,AC0                 ;MOVE REGISTER CONTENTS
10585 051516  174067  131366                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10586 051522  004767  000060                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10587 051526  001401                          BEQ     MBT8F                   ;BRANCH IF GOOD
10588 051530  104003                          ERROR   .3                      ;FPP ERROR
10589                                                                         ;BAD AC5
10590 051532  005204          MBT8F:  INC     R4                              ;INCREMENT PATTERN COUNTER
10591 051534  000241                          CLC
10592 051536  042704  177776                  BIC     #177776,R4              ; TEST FOR ODD /EVEN
10593 051542  001401                          BEQ     MBT8FG                  ;BRANCH IF EVEN
10594 051544  000261                          SEC                             ;SET CARRY FOR TEST PATTERN SHIFT
10595 051546  006112          MBT8FG: ROL     (R2)                            ;ROTATE LSW OF TEST PATTERN
10596 051550  006162  000002                  ROL     2(R2)                   ;ROTATE 2 WORD OF TEST PATTERN
10597 051554  006162  000004                  ROL     4(R2)                   ;ROTATE 3 WORD OF TEST PATTERN
10598 051560  006162  000006                  ROL     6(R2)                   ;ROTATE 4 WORD OF TEST PATTERN
10599 051564  103402                          BCS     MBT8I                   ;JUMP IF THROUGH WITH TEST PATTERN
10600 051566  000167  177350                  JMP     BTGO                    ;CONTINUE WITH NEW TEST PATTERN
10601
10602 051572  005205          MBT8I:  INC     R5              ;GO TO NEXT REGISTER TEST
10603 051574  020527  000006                  CMP     R5,#6           ;SEE IF THROUGH TESTING
10604 051600  100016                          BPL     MBTE            ;JUMP IF THROUGH
10605 051602  000167  177326                  JMP     MBT2A           ;CONTINUE TESTING WITH NEW PATTERN
10606                                 ;
10607                                 ;
10608 051606  005767  131276          BTTST:  TST     BTRES           ;VERIFY CONTENTS AS ZERO
10609 051612  001010                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10610 051614  005767  131272                  TST     BTRES+2         ;VERIFY CONTENTS AS ZERO
10611 051620  001005                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10612 051622  005767  131266                  TST     BTRES+4         ;VERIFY CONTENTS AS ZERO
10613 051626  001002                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10614 051630  005767  131262                  TST     BTRES+6         ;VERIFY CONTENTS AS ZERO
10615 051634  000207          BTTSTE: RTS     R7              ;GO BACK TO CALLING ROUTINE
10616
10617
10618                                         ;
10619                                         ;
10620 051636                  MBTE:
10621
10623                                 ;
10634 051636                  MFACU:
10635
10636                                 ;       TEST UNIQUENESS OF FPP ACCUMULATORS
10637                                 ;;**************************************************************************
                                      ;THIS TEST LOADS UNIQUE PATTERNS INTO EACH ACCUMULATOR SIMULTANEOUSLY.
                                      ;R2=POINTER TO EXPECTED DATA
                                      ;R3=POINTER TO RECEIVED DATA
10638                                           ;
```

FLOATING POINT TESTS

```
10639                                                   ;
10640                                                   ;
10641 051636  170011                                    SETD
10642 051640  005000                            MFA:    CLR     R0                      ;SETUP FPP ACC POINTER
10643 051642  005004                                    CLR     R4
10644 051644  012702  003100                            MOV     #BTEXP,R2               ;POINT TO TEST DATA
10645 051650  012703  003110                            MOV     #BTRES,R3               ;POINT TO RECEIVED DATA
10646 051654  012767  000051  131216                    MOV     #51,BTEXP               ;SETUP EXPECTED DATA
10647 051662  012767  000052  131212                    MOV     #52,BTEXP+2             ;
10648 051670  012767  000053  131206                    MOV     #53,BTEXP+4             ;
10649 051676  012767  000054  131202                    MOV     #54,BTEXP+6             ;
10650 051704  172467  131170                            LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10651 051710  174005                                    STD     AC0,AC5                 ;PUT DATA INTO TEST REGISTER
10652 051712  004567  000210                            JSR     R5,SUBT                 ;SUBTRACT TEN FROM EACH EXPECTED DATA
10653 051716  172467  131156                            LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10654 051722  174004                                    STD     AC0,AC4                 ;MOVE DATA INTO TEST REGISTER
10655 051724  004567  000176                            JSR     R5,SUBT                 ;SUBTRACT 10 FROM TEST DATA WORDS
10656 051730  172767  131144                            LDD     BTEXP,AC3               ;STORE INTO TEST REGISTER
10657 051734  004567  000166                            JSR     R5,SUBT                 ;GET NEXT SET OF UNIQUE DATA WORDS
10658 051740  172667  131134                            LDD     BTEXP,AC2               ;STORE INTO TEST REGISTER
10659 051744  004567  000156                            JSR     R5,SUBT                 ;GET NEXT SET OF TEST DATAS
10660 051750  172567  131124                            LDD     BTEXP,AC1               ;LOAD TEST REGISTER
10661 051754  004567  000146                            JSR     R5,SUBT                 ;GET NEXT SET OF TEST WORDS
10662 051760  172467  131114                            LDD     BTEXP,AC0               ;LOAD FINAL TEST REGISTER
10663                                                                                   ;ALL REGISTER CONTAIN UNIQUE TEST WORDS
10664 051764  174067  131120                            STD     AC0,BTRES               ;STORE AC0,RESULT
10665 051770  004567  000216                            JSR     R5,BFA                  ;CHECK RESULT
10666 051774  001401                                    BEQ     BFAC1                   ;BRANCH IF GOOD
10667 051776  104003                                    ERROR   +3              ;FPP ERROR
10668                                                                           ;BAD AC0
10669 052000  004567  000154                    BFAC1:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10670 052004  174167  131100                            STD     AC1,BTRES               ;STORE AC1 RESULT
10671 052010  004567  000176                            JSR     R5,BFA                  ;CHECK RESULT
10672 052014  001401                                    BEQ     BFAC2                   ;BRANCH IF GOOD
10673 052016  104003                                    ERROR   +3              ;FPP ERROR
10674                                                                           ;BAD RESULT AC1
10675 052020  004567  000134                    BFAC2:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10676 052024  174267  131060                            STD     AC2,BTRES               ;STORE AC2 RESULT
10677 052030  004567  000156                            JSR     R5,BFA                  ;CHECK RESULT
10678 052034  001401                                    BEQ     BFAC3                   ;BRANCH IF GOOD
10679 052036  104003                                    ERROP   +3              ;FPP ERROR
10680                                                                           ;BAD AC2 RESLUT
10681 052040  004567  000114                    BFAC3:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10682 052044  174367  131040                            STD     AC3,BTRES               ;SAVE TEST RESULT
10683 052050  004567  000136                            JSR     R5,BFA                  ;CHECK RESULT
10684 052054  001401                                    BEQ     BFAC4                   ;BRANCH IF GOOD
10685 052056  104003                                    ERROR   +3              ;FPP ERROR
10686                                                                           ;BAD AC3 RESULT
10687 052060  004567  000074                    BFAC4:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10688 052064  172704                                    LDD     AC4,AC3                 ;SAVE TEMPORARY
10689 052066  174367  131016                            STD     AC3,BTRES               ;STORE AC4 RESULT
10690 052072  004567  000114                            JSR     R5,BFA                  ;CHECK RESULT
10691 052076  001401                                    BEQ     BFAC5                   ;BRANCH IF GOOD
10692 052100  104003                                    ERROR   +3              ;FPP ERROR
10693                                                                           ;BAD AC4 RESULT
10694 052102  004567  000052                    BFAC5:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10695 052106  172605                                    LDD     AC5,AC2                 ;SAVE TEMPORARY COPY
```

FLOATING POINT TESTS

```
10696 052110  174267  130774                STD     AC2,BTRES               ;MOVE AC5 RESULT
10697 052114  004567  000072                JSR     R5,BFA                  ;CHECK RESULT
10698 052120  001454                        BEQ     BFAE                    ;BRANCH IF GOOD
10699 052122  104003                        ERROR   +3              ;FPP ERROR
10700                                                               ;BAD AC5 RESULT
10701 052124  000452                        BR      BFAE                    ;EXIT MODULE
10702
10703 052126  162767  000010  130744  SUBT: SUB     #10,BTEXP                       ;UPDATE EXPECTED CONTENTS
10704 052134  162767  000010  130740        SUB     #10,BTEXP+2                     ;UPDATE EXPECTED CONTENTS
10705 052142  162767  000010  130734        SUB     #10,BTEXP+4                     ;UPDATE EXPECTED CONTENTS
10706 052150  162767  000010  130730        SUB     #10,BTEXP+6                     ;UPDATE EXPECTED CONTENTS
10707 052156  000205                        RTS     R5                      ;
10708 052160  062767  000010  130712  ADDT: ADD     #10,BTEXP                       ;UPDATE EXPECTED CONTENTS
10709 052166  062767  000010  130706        ADD     #10,BTEXP+2                     ;UPDATE EXPECTED CONTENTS
10710 052174  062767  000010  130702        ADD     #10,BTEXP+4                     ;UPDATE EXPECTED CONTENTS
10711 052202  062767  000010  130676        ADD     #10,BTEXP+6                     ;UPDATE EXPECTED CONTENTS
10712 052210  000205                        RTS     R5                      ;
10713
10714 052212  026767  130662  130670  BFA:  CMP     BTEXP,BTRES             ;VERIFY CONTENTS
10715 052220  001013                        BNE     BFB                     ;EXIT IF NOT ZERO
10716 052222  026767  130654  130662        CMP     BTEXP+2,BTRES+2         ;VERIFY CONTENTS
10717 052230  001007                        BNE     BFB                     ;EXIT IF NOT ZERO
10718 052232  026767  130646  130654        CMP     BTEXP+4,BTRES+4         ;VERIFY CONTENTS
10719 052240  001003                        BNE     BFB                     ;EXIT IF NOT ZERO
10720 052242  026767  130640  130646        CMP     BTEXP+6,BTRES+6         ;VERIFY CONTENTS
10721 052250  000205                  BFB:  RTS     R5                      ;GO BACK TO CALLING ROUTINE
10722
10723
10724 052252                          BFAE:
10725
10726
10727
10730 052252                          TSFP1:
10731                                 ;       TEST LDFPS AND STFPS MODE 0
10732 052252  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10733 052256  012704  147757                MOV     #147757,R4              ;SETUP DATA TO BE LOADED
10734 052262  004767  000032                JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10735 052266  012704  105252                MOV     #105252,R4              ;SETUP DATA TO BE LOADED
10736 052272  004767  000022                JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10737 052276  012704  042505                MOV     #42505,R4               ;SETUP DATA TO BE LOADED
10738 052302  004767  000012                JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10739 052306  005004                        CLR     R4                      ;SETUP DATA TO BE LOADED
10740 052310  004767  000004                JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10741
10742                                 ;
10743 052314  000167  000014                JMP     FIN1
10744                                 ;
10745 052320  170104                  LOST: LDFPS   R4                      ;LOAD FPS WITH DATA
10746 052322  170201                        STFPS   R1                      ;LOAD R1 WITH (FPS)
10747 052324  020401                        CMP     R4,R1                   ;DID THE INSTRUCTIONS WORK
10748 052326  001401                        BEQ     1$                      ;YES GO ON
10749 052330  104003                        ERROR   +3              ;FPP ERROR
10750                                                               ;NO GO TO ERROR
10751 052332  000207                  1$:   RTS     PC                      ;RETURN
10752 052334  000240                  FIN1: NOP
10753                                 ;
10756 052336                          TSFP2:
```

FLOATING POINT TESTS

```
10757                                   ;       TEST CFCC
10758 052336  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10759 052342  012704  000017                    MOV     #17,R4                  ;SETUP DATA TO BE LOADED
10760 052346  004767  000032                    JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10761 052352  012704  000012                    MOV     #12,R4                  ;SETUP DATA TO BE LOADED
10762 052356  004767  000022                    JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10763 052362  012704  000005                    MOV     #5,R4                   ;SETUP DATA TO BE LOADED
10764 052366  004767  000012                    JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10765 052372  005004                            CLR     R4                      ;SETUP DATA TO BE LOADED
10766 052374  004767  000004                    JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10767
10768                                   ;
10769 052400  000167  000024                    JMP     FIN2
10770                                   ;
10771 052404  170104                    TSF2:   LDFPS   R4                      ;LOAD FPS
10772 052406  170000                            CFCC                            ;COPY CONDITION CODES TO PS
10773 052410  013701  177776                    MOV     @#177776,R1             ;SAVE PS TO R1
10774 052414  042701  177760                    BIC     #177760,R1              ;MASK OUT UNWANTED BITS
10775 052420  020401                            CMP     R4,R1                   ;WAS CONDITION CODE BITS TRANSFERRED
10776                                                                           ;CORRECTLY
10777 052422  001401                            BEQ     1$                      ;YES GO ON
10778 052424  104003                            ERROR   .+3             ;FPP ERROR
10779                                                                   ;NO GO TO ERROR
10780 052426  000207                    1$:     RTS     PC                      ;RETURN
10781 052430                            FIN2:
10782                                   ;
10785 052430                            TSFP3:
10786                                   ;       TEST SETF, SETD, SETI, SETL
10787 052430  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10788 052434  012704  000200                    MOV     #200,R4                 ;SETUP DATA TO BE LOADED
10789 052440  170104                            LDFPS   R4                      ;LOAD FPS
10790 052442  170001                            SETF                            ;MAKE FD=0
10791 052444  170201                            STFPS   R1                      ;STORE FPS
10792 052446  020127  000000                    CMP     R1,#0                   ;IS FD=0
10793 052452  001401                            BEQ     1$                      ;YES GO ON
10794 052454  104003                            ERROR   .+3             ;FPP ERROR
10795                                                                   ;NO GO TO ERROR
10796 052456  170011                    1$:     SETD                            ;MAKE FD=1
10797 052460  170201                            STFPS   R1                      ;STORE FPS
10798 052462  020104                            CMP     R1,R4                   ;IS FD=1
10799 052464  001401                            BEQ     2$                      ;YES GO ON
10800 052466  104003                            ERROR   .+3             ;FPP ERROR
10801                                                                   ;NO GO TO ERROR
10802 052470  012704  000100            2$:     MOV     #100,R4                 ;SETUP DATA TO BE LOADED
10803 052474  170104                            LDFPS   R4                      ;LOAD FPS
10804 052476  170002                            SETI                            ;MAKE FL=0
10805 052500  170201                            STFPS   R1                      ;STORE FPS
10806 052502  020127  000000                    CMP     R1,#0                   ;IS FL=0
10807 052506  001401                            BEQ     3$                      ;YES GO ON
10808 052510  104003                            ERROR   .+3             ;FPP ERROR
10809                                                                   ;NO GO TO ERROR
10810 052512  170012                    3$:     SETL                            ;MAKE FL=1
10811 052514  170201                            STFPS   R1                      ;STORE FPS
10812 052516  020104                            CMP     R1,R4                   ;IS FL=1
10813 052520  001401                            BEQ     4$                      ;YES GO ON
10814 052522  104003                            ERROR   .+3             ;FPP ERROR
10815                                                                   ;NO GO TO ERROR
```

FLOATING POINT TESTS

```
10816 052524                          4$:
10817                                 ;
10820 052524                          TSFP4:
10821                                 ;       TEST ILLEGAL OP CODES AND STST
10822 052524  005037  135370                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10823 052530  012705  170003                  MOV     #170003,R5              ;INIT OP CODE
10824 052534  013746  000244                  MOV     @#244,-(SP)             ;SAVE FP VECTOR
10825 052540  012737  052670  000244          MOV     #ILLOP1,@#244           ;SETUP NEW VECTOR
10826 052546  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
10827 052552  012737  052740  000004          MOV     #TIMEOU,@#4             ;SETUP NEW VECTOR
10828 052560  013746  000010                  MOV     @#10,-(SP)              ;SAVE ILLEGAL VECTOR
10829 052564  012737  052744  000010          MOV     #ILLOP2,@#10            ;SETUP NEW VECTOR
10830 052572  005003                  D1:     CLR     R3                      ;
10831 052574  170103                          LDFPS   R3                      ;CLEAR FPS
10832 052576  005002                          CLR     R2                      ;
10833 052600  010537  052604                  MOV     R5,@#D2                 ;SETUP THE ILLEGAL INST
10834 052604  000000                  D2:     .WORD   0                       ;
10835 052606  170000                  D3:     CFCC                            ;MEMORY WORDS TO BE USED WITH
10836 052610  005202                          INC     R2                      ;EXECUTION OF ILLEGAL OP CODE
10837 052612  005202                          INC     R2                      ;
10838 052614  170201                          STFPS   R1                      ;SAVE FPS
10839 052616  104003                          ERROR   +3              ;FPP ERROR
10840                                                                 ;GO TO ERROR
10841 052620  022705  170010          D4:     CMP     #170010,R5              ;COMPUTE NEXT OP CODE
10842 052624  001003                          BNE     D5                      ;
10843 052626  012705  170013                  MOV     #170013,R5              ;
10844 052632  000757                          BR      D1                      ;
10845 052634  022705  170077          D5:     CMP     #170077,R5              ;
10846 052640  001001                          BNE     D6                      ;
10847 052642  000402                          BR      D7                      ;
10848 052644  005205                  D6:     INC     R5                      ;
10849 052646  000751                          BR      D1                      ;
10850 052650  012637  000010          D7:     MOV     (SP)+,@#10              ;RESTORE VECTORS
10851 052654  012637  000004                  MOV     (SP)+,@#4               ;
10852 052660  012637  000244                  MOV     (SP)+,@#244             ;
10853
10854                                 ;
10855 052664  000167  000060                  JMP     FIN4
10856                                 ;
10857 052670  022716  052606          ILLOP1: CMP     #D3,(SP)                ;DID TRAP OCCUR ON TEST INST
10858 052674  001401                          BEQ     1$                      ;YES GO ON
10859 052676  104003                          ERROR   +3              ;FPP ERROR
10860                                                                 ;NO GO TO ERROR
10861 052700  022626                  1$:     CMP     (SP)+,(SP)+             ;CLEAN UP STACK
10862 052702  170201                          STFPS   R1                      ;STORE FPS
10863 052704  022701  100000                  CMP     #100000,R1              ;IS FPS CORRECT
10864 052710  001401                          BEQ     2$                      ;YES GO ON
10865 052712  104003                          ERROR   +3              ;FPP ERROR
10866                                                                 ;NO GO TO ERROR
10867 052714  005004                  2$:     CLR     R4                      ;INT R4 TO A KNOWN STATE
10868 052716  170304                          STST    R4                      ;STORE FEC AT R4
10869                                                                 ;IF THE DESTINATION MODE IS IMPROPERLY
10870                                                                 ;DECODED AN ODD ADDRESS TRAP TO 4
10871                                                                 ;SHOULD OCCUR
10872 052720  022704  000002                  CMP     #2,R4                   ;IS FEC CORRECT
10873 052724  001002                          BNE     3$                      ;NO GO TO ERROR
10874 052726  000167  177666                  JMP     D4                      ;YES GO ON
```

FLOATING POINT TESTS

```
10875 052732  104003                 3$:     ERROR   .3              ;FPP ERROR
10876                                                                 ;GO TO ERROR
10877 052734  000167  177660                  JMP     D4                      ;THEN GO ON
10878                               ;
10879 052740  104003                 TIMEOU: ERROR   .3              ;FPP ERROR
10880                                                                 ;ERROR BECAUSE OF TRAP TO 4
10881 052742  000006                          RTT                             ;RETURN
10882                               ;
10883 052744  104003                 ILLOP2: ERROR   .3              ;FPP ERROR
10884                                                                 ;ERROR BECAUSE OF TRAP TO 10
10885 052746  000006                          RTT                             ;RETURN
10886 052750                         FIN4:
10887                               ;
10890 052750                         TSFP5:
10891                                ;       TEST FID (INTERRUPT DISABLE BIT)
10892 052750  005037  135370                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10893 052754  013746  000244                  MOV     @#244,-(SP)             ;SAVE FP VECTOR
10894 052760  012737  053034  000244          MOV     #ILL,@#244              ;SETUP NEW VECTOR
10895 052766  012703  040000                  MOV     #40000,R3               ;SETUP DATA TO BE LOADED
10896 052772  170103                          LDFPS   R3                      ;LOAD FPS, FID=1
10897 052774  170020                          .WORD   170020                  ;ILLEGAL FP INSTRUCTION
10898 052776  170000                          CFCC
10899 053000  170201                          STFPS   R1                      ;SEE IF ERROR WAS RECORDED IN FPS
10900 053002  022701  140000                  CMP     #140000,R1              ;
10901 053006  001401                          BEQ     1$                      ;YES GO ON
10902 053010  104003                          ERROR   .3              ;FPP ERROR
10903                                                                 ;NO GO TO ERROR
10904 053012  170304                 1$:     STST    R4                      ;SEE IF FEC=2
10905 053014  022704  000002                  CMP     #2,R4                   ;
10906 053020  001401                          BEQ     2$                      ;YES GO ON
10907 053022  104003                          ERROR   .3              ;FPP ERROR
10908                                                                 ;NO GO TO ERROR
10909 053024  012637  000244         2$:     MOV     (SP)+,@#244             ;RESTORE VECTOR
10910
10911                                ;
10912 053030  000167  000004                  JMP     FIN5
10913                               ;
10914 053034  104003                 ILL:    ERROR   .3              ;FPP ERROR
10915                                                                 ;FID ERROR
10916 053036  000006                          RTT                             ;RETURN
10917 053040                         FIN5:
10918                               ;
10921 053040                         TSFP6:
10922                                ;       TEST LDD, STD FSRC AND FDST MODE 1
10923 053040  005037  135370                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10924 053044  005004                          CLR     R4                      ;SETUP TO LOAD DATA
10925 053046  170104                          LDFPS   R4                      ;CLEAR FPS
10926 053050  170011                          SETD                            ;SET FD TO 1
10927 053052  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10928 053056  012737  053210  000004          MOV     #TSF6,@#4               ;SETUP NEW VECTOR
10929 053064  012704  053200                  MOV     #TS6DAT,R4              ;SETUP POINTER TO DATA
10930 053070  172414                          LDD     (R4),AC0                ; TEST INSTRUCTION
10931 053072  020427  053200                  CMP     R4,#TS6DAT              ;IS R4 CORRECT
10932 053076  001401                          BEQ     1$                      ;YES GO ON
10933 053100  104003                          ERROR   .3              ;FPP ERROR
10934                                                                 ;NO GO TO ERROR
10935 053102  012701  053170         1$:     MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
```

FLOATING POINT TESTS

```
10936 053106  012703  000004          MOV     #4,R3                   ;INIT COUNTER
10937 053112  022421            2$:   CMP     (R4)+,(R1)+             ;WAS SOURCE DATA ALTERED
10938 053114  001401                  BEQ     3$                      ;NO GO ON
.10939 053116  104003                 ERROR   +3              ;FPP ERROR
10940                                         ;YES GO TO ERROR
10941 053120  077304            3$:   SOB     R3,2$                   ;ARE WE DONE
10942 053122  012704  003162          MOV     #TSTLOC,R4              ;SETUP POINTER FOR DATA
10943 053126  174014                  STD     AC0,(R4)                ; TEST INSTRUCTION
10944 053130  020427  003162          CMP     R4,#TSTLOC                      ;IS R4 CORRECT
10945 053134  001401                  BEQ     4$                      ;YES GO ON
10946 053136  104003                  ERROR   +3              ;FPP ERROR
10947                                         ;NO GO TO ERROR
10948 053140  012701  053170    4$:   MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
10949 053144  012703  000004          MOV     #4,R3                   ;INIT COUNTER
10950 053150  022421            5$:   CMP     (R4)+,(R1)+             ;IS DESTINATION DATA CORRECT
10951 053152  001401                  BEQ     6$                      ;YES GO ON
10952 053154  104003                  ERROR   +3              ;FPP ERROR
10953                                         ;NO GO TO ERROR
10954 053156  077304            6$:   SOB     R3,5$                   ;ARE WE DONE
10955 053160  012637  000004          MOV     (SP)+,@#4               ;RESTORE VECTOR
10956
10957                           ;
10958 053164  000167  000024          JMP     FIN6
10959                           ;
10960 053170  177777            TS6DA: .WORD  177777
10961 053172  000000                   .WORD  000000
10962 053174  052525                   .WORD  052525
10963 053176  125252                   .WORD  125252
10964 053200  177777            TS6DAT: .WORD 177777
10965 053202  000000                   .WORD  000000
10966 053204  052525                   .WORD  052525
10967 053206  125252                   .WORD  125252
10968                           ;
10969 053210  104003            TSF6:  ERROR  +3                ;FPP ERROR
10970                                          ;ODD ADDRESS TRAP
10971 053212  000006                   RTT                             ;RETURN
10972 053214                    FIN6:
10973                           ;
10976 053214                    TSFP7:
10977                           ;       TEST LDD, LDF FSRC MODE 0
10978 053214  005037  135370          CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10979 053220  012704  000200          MOV     #200,R4                 ;SETUP TO LOAD FPS
10980 053224  170104                  LDFPS   R4                      ;LOAD FPS, FD=1
10981 053226  013746  000004          MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
10982 053232  012737  053400  000004  MOV     #TSF7,@#4               ;SETUP NEW VECTOR
10983 053240  012704  053404          MOV     #TS7DA1,R4              ;SETUP POINTER TO DATA
10984 053244  172414                  LDD     (R4),AC0                ;CLEAR AC0
10985 053246  012701  053414          MOV     #TS7DA2,R1              ;SETUP POINTER TO DATA
10986 053252  172511                  LDD     (R1),AC1                ;LOAD AC1 WITH DATA
10987 053254  172401                  LDD     AC1,AC0                 ; TEST INSTRUCTION
10988 053256  012704  003162          MOV     #TSTLOC,R4              ;
10989 053262  174114                  STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
10990 053264  004767  000072          JSR     PC,CHECK7               ;
10991 053270  012704  003162          MOV     #TSTLOC,R4                      ;SETUP POINTERS FOR DATA
10992 053274  012701  053414          MOV     #TS7DA2,R1              ;
10993 053300  174014                  STD     AC0,(R4)                ;CHECK IF AC0 RECEIVED CORRECT DATA
10994 053302  004767  000054          JSR     PC,CHECK7               ;
```

FLOATING POINT TESTS

```
10995 053306  012701  053404                MOV     #TS7DA1,R1              ;SETUP POINTER TO DATA
10996 053312  172511                       LDD     (R1),AC1                ;CLEAR AC1
10997 053314  170001                       SETF                            ;SET FD=0
10998 053316  172401                       LDF     AC1,AC0                 ; TEST INSTRUCTION
10999 053320  170011                       SETD                            ;SET FD=1
11000 053322  012704  003162               MOV     #TSTLOC,R4              ;SETUP POINTER TO DATA
11001 053326  174114                       STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
11002 053330  004767  000026               JSR     PC,CHECK7               ;
11003 053334  012704  053424               MOV     #TS7DA4,R4              ;SETUP POINTERS FOR DATA
11004 053340  012701  003162               MOV     #TSTLOC,R1              ;
11005 053344  174011                       STD     AC0,(R1)                ;CHECK IF AC0 HAS CORRECT DATA
11006 053346  004767  000010               JSR     PC,CHECK7               ;
11007 053352  012637  000004               MOV     (SP)+,@#4               ;RESTORE VECTOR
11008
11009                               ;
11010 053356  000167  000052               JMP     FIN7
11011                               ;
11012 053362  012703  000004        CHECK7: MOV     #4,R3                   ;INIT COUNTER
11013 053366  022421                CHEK7:  CMP     (R4)+,(R1)+             ;IS DATA OK
11014 053370  001401                        BEQ     CHK7                    ;YES GO ON
11015 053372  104003                        ERROR   .3                  ;FPP ERROR
11016                                                                   ;NO GO TO ERROR
11017 053374  077304                CHK7:   SOB     R3,CHEK7                ;ARE WE DONE
11018 053376  000207                        RTS     PC                      ;YES RETURN
11019                               ;
11020 053400  104003                TSF7:   ERROR   .3                  ;FPP ERROR
11021                                                                   ;ODD ADDRESS TRAP
11022 053402  000006                        RTT                             ;RETURN
11023                               ;
11024 053404  000000                TS7DA1: .WORD   0
11025 053406  000000                        .WORD   0
11026 053410  000000                        .WORD   0
11027 053412  000000                        .WORD   0
11028 053414  037641                TS7DA2: .WORD   37641
11029 053416  065121                        .WORD   65121
11030 053420  037373                        .WORD   37373
11031 053422  022265                        .WORD   22265
11032 053424  000000                TS7DA4: .WORD   0
11033 053426  000000                        .WORD   0
11034 053430  037373                        .WORD   37373
11035 053432  022265                        .WORD   22265
11036 053434                        FIN7:
11037                               ;
11040 053434                        TSFP10:
11041                               ;       TEST STD, STF FDST MODE 0
11042 053434  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11043 053440  012704  000200                MOV     #200,R4                 ;SETUP TO LOAD FPS
11044 053444  170104                        LDFPS   R4                      ;LOAD FPS, FD=1
11045 053446  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11046 053452  012737  053620  000004        MOV     #TSF10,@#4              ;SETUP NEW VECTOR
11047 053460  012704  053624                MOV     #TS10D1,R4              ;SETUP POINTER TO DATA
11048 053464  172414                        LDD     (R4),AC0                ;CLEAR AC0
11049 053466  012701  053634                MOV     #TS10D2,R1              ;SETUP POINTER TO DATA
11050 053472  172511                        LDD     (R1),AC1                ;LOAD AC1 WITH DATA
11051 053474  174100                        STD     AC1,AC0                 ; TEST INSTRUCTION
11052 053476  012704  003162                MOV     #TSTLOC,R4              ;
11053 053502  174114                        STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
```

FLOATING POINT TESTS

```
11054 053504  004767  000072                JSR     PC,CHEC10           ;
11055 053510  012704  003162                MOV     #TSTLOC,R4          ;SETUP POINTERS FOR DATA
11056 053514  012701  053634                MOV     #TS10D2,R1          ;
11057 053520  174014                        STD     AC0,(R4)            ;CHECK IF AC0 RECEIVED CORRECT DATA
11058 053522  004767  000054                JSR     PC,CHEC10           ;
11059 053526  012701  053624                MOV     #TS10D1,R1          ;SETUP POINTER TO DATA
11060 053532  172511                        LDD     (R1),AC1            ;CLEAR AC1
11061 053534  170001                        SETF                        ;SET FD=0
11062 053536  174100                        STF     AC1,AC0             ; TEST INSTRUCTION
11063 053540  170011                        SETD                        ;SET FD=1
11064 053542  012704  003162                MOV     #TSTLOC,R4          ;SETUP POINTER TO DATA
11065 053546  174114                        STD     AC1,(R4)            ;CHECK IF AC1 HAS BEEN ALTERED
11066 053550  004767  000026                JSR     PC,CHEC10           ;
11067 053554  012704  053644                MOV     #TS10D4,R4          ;SETUP POINTERS FOR DATA
11068 053560  012701  003162                MOV     #TSTLOC,R1          ;
11069 053564  174011                        STD     AC0,(R1)            ;CHECK IF AC0 HAS CORRECT DATA
11070 053566  004767  000010                JSR     PC,CHEC10           ;
11071 053572  012637  000004                MOV     (SP)+,@#4           ;RESTORE VECTOR
11072
11073                               ;
11074 053576  000167  000052                JMP     FIN10
11075                               ;
11076 053602  012703  000004      CHEC10:   MOV     #4,R3               ;INIT COUNTER
11077 053606  022421              CH10:     CMP     (R4)+,(R1)+         ;IS DATA OK
11078 053610  001401                        BEQ     CHK10               ;YES GO ON
11079 053612  104003                        ERROR   +3          ;FPP ERROR
11080                                                           ;NO GO TO ERROR
11081 053614  077304              CHK10:    SOB     R3,CH10             ;ARE WE DONE
11082 053616  000207                        RTS     PC                  ;YES RETURN
11083                               ;
11084 053620  104005              TSF10:    ERROR   +3          ;FPP ERROR
11085                                                           ;ODD ADDRESS TRAP
11086 053622  000006                        RTT                         ;
11087                               ;
11088 053624  000000              TS10D1:   .WORD   0
11089 053626  000000                        .WORD   0
11090 053630  000000                        .WORD   0
11091 053632  000000                        .WORD   0
11092 053634  177777              TS10D2:   .WORD   177777
11093 053636  111236                        .WORD   111236
11094 053640  100045                        .WORD   100045
11095 053642  003651                        .WORD   3651
11096 053644  000000              TS10D4:   .WORD   0
11097 053646  000000                        .WORD   0
11098 053650  100045                        .WORD   100045
11099 053652  003651                        .WORD   3651
11100 053654                      FIN10:
11101                               ;
11104 053654                      TSFP11:
11105                             ;         TEST FDST SINGLE OPERAND MODE 0
11106 053654  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11107 053660  012704  000200                MOV     #200,R4                 ;SETUP TO LOAD FPS
11108 053664  170104                        LDFPS   R4                      ;SET FD=1
11109 053666  012704  053740                MOV     #TS11D1,R4              ;SETUP POINTER TO DATA
11110 053672  172414                        LDD     (R4),AC0                ;LOAD ALL ONES TO AC0
11111 053674  170400                        CLRD    AC0                     ; TEST INSTRUCTION
11112 053676  170203                        STFPS   R3                      ;GET FPS
```

FLOATING POINT TESTS

```
11113 053700  012704  003162                MOV     #TSTLOC,R4              ;
11114 053704  174014                        STD     AC0,(R4)                ;CHECK AC0 FOR ALL ZEROES
11115 053706  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11116 053712  022427  000000        1$:     CMP     (R4)+,#0                ;
11117 053716  001401                        BEQ     2$                      ;OK GO ON
11118 053720  104003                        ERROR   .3              ;FPP ERROR
11119                                                               ;NO GO TO ERROR
11120 053722  077105                2$:     SOB     R1,1$                   ;ARE WE DONE
11121 053724  020327  000204                CMP     R3,#204                 ;CHECK FPS
11122 053730  001401                        BEQ     3$                      ;OK GO ON
11123 053732  104003                        ERROR   .3              ;FPP ERROR
11124                                                               ;NO GO TO ERROR
11125 053734                        3$:
11126                               ;
11127 053734  000167  000010                JMP     FIN11
11128                               ;
11129 053740  177777                TS11D1: .WORD   177777
11130 053742  177777                        .WORD   177777
11131 053744  177777                        .WORD   177777
11132 053746  177777                        .WORD   177777
11133 053750                        FIN11:
11134                               ;
11137 053750                        TSFP12:
11138                               ;       TEST FDST SOP MODE 0 WITH ILLEGAL AC7
11139 053750  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11140 053754  012703  040200                MOV     #40200,R3               ;SETUP TO LOAD FPS
11141 053760  170103                        LDFPS   R3                      ;SET FID=1, AND FD=1
11142 053762  170407                        CLRD    AC7                     ; TEST INSTRUCTION
11143 053764  170204                        STFPS   R4                      ;GET FPS
11144 053766  170305                        STST    R5                      ;GET FEC
11145 053770  022704  140200                CMP     #140200,R4              ;IS FPS CORRECT
11146 053774  001401                        BEQ     1$                      ;YES GO ON
11147 053776  104003                        ERROR   .3              ;FPP ERROR
11148                                                               ;NO GO TO ERROR
11149 054000  022705  000002        1$:     CMP     #2,R5                   ;IS FEC CORRECT
11150 054004  001401                        BEQ     2$                      ;YES GO ON
11151 054006  104003                        ERROR   .3              ;FPP ERROR
11152                                                               ;NO GO TO ERROR
11153 054010                        2$:
11154                               ;
11157 054010                        TSFP13:
11158                               ;       TEST FDST SOP MODE 1
11159 054010  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11160 054014  012737  054130  000004        MOV     #TSF13,@#4              ;SETUP NEW VECTOR
11161 054022  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11162 054026  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11163 054032  170102                        LDFPS   R2                      ;SET FD=1
11164 054034  012705  000004                MOV     #4,R5                   ;INIT COUNTER
11165 054040  012704  003162                MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11166 054044  012724  177777        100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
11167 054050  077503                        SOB     R5,100$                 ;ARE WE DONE
11168 054052  012702  003162                MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11169 054056  170412                        CLRD    (R2)                    ; TEST INSTRUCTION
11170 054060  170203                        STFPS   R3                      ;GET FPS
11171 054062  020227  003162                CMP     R2,#TSTLOC                      ;WAS R2 ALTERED
11172 054066  001401                        BEQ     1$                      ;NO GO ON
11173 054070  104003                        ERROR   .3              ;FPP ERROR
```

FLOATING POINT TESTS

```
11174
11175 054072  012701  000004          1$:     MOV     #4,R1                   ;YES GO TO ERROR
11176 054076  022227  000000          2$:     CMP     (R2)+,#0                ;INIT COUNTER
11177 054102  001401                          BEQ     3$                      ;CHECK LOCATION FOR 0
11178 054104  104003                          ERROR   .3                      ;OK GO ON
11179                                                                         ;FPP ERROR
11180 054106  077105                  3$:     SOB     R1,2$                   ;NO GO TO ERROR
11181 054110  020327  000204                    CMP     R3,#0204                ;ARE WE DONE
11182 054114  001401                          BEQ     4$                      ;CHECK FPS
11183 054116  104003                          ERROR   .3                      ;OK GO ON
11184                                                                         ;FPP ERROR
11185 054120  012637  000004          4$:     MOV     (SP)+,@#4               ;NO GO TO ERROR
11186                                                                         ;RESTORE VECTOR
11187                                 ;
11188 054124  000167  000004                    JMP     FIN13
11189                                 ;
11190 054130  104003                  TSF13:  ERROR   .3                      ;FPP ERROR
11191                                                                         ;ODD ADDRESS TRAP
11192 054132  000006                          RTT                             ;RETURN
11193                                 ;
11194 054134                          FIN13:
11195                                 ;
11198 054134                          TSFP14:
11199                                 ;       TEST FDST SOP MODE 2
11200 054134  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11201 054140  012737  054260  000004          MOV     #TSF14,@#4              ;SETUP NEW VECTOR
11202 054146  005037  135370                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11203 054152  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11204 054156  170102                          LDFPS   R2                      ;SET FD=1
11205 054160  012705  000004                  MOV     #4,R5                   ;INIT COUNTER
11206 054164  012704  003162                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11207 054170  012724  177777          100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
11208 054174  077503                          SOB     R5,100$                 ;ARE WE DONE
11209 054176  012702  003162                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11210 054202  170422                          CLRD    (R2)+                   ; TEST INSTRUCTION
11211 054204  170203                          STFPS   R3                      ;GET FPS
11212 054206  020227  003172                  CMP     R2,#TSTLOC+10                   ;IS R2 CORRECT
11213 054212  001401                          BEQ     1$                      ;YES GO ON
11214 054214  104003                          ERROR   .3                      ;FPP ERROR
11215                                                                         ;NO GO TO ERROR
11216 054216  012702  003162          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11217 054222  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11218 054226  022227  000000          2$:     CMP     (R2)+,#0                ;CHECK LOCATION FOR 0
11219 054232  001401                          BEQ     3$                      ;YES GO ON
11220 054234  104003                          ERROR   .3                      ;FPP ERROR
11221                                                                         ;NO GO TO ERROR
11222 054236  077105                  3$:     SOB     R1,2$                   ;ARE WE DONE
11223 054240  020327  000204                  CMP     R3,#0204                ;CHECK FPS
11224 054244  001401                          BEQ     4$                      ;OK GO ON
11225 054246  104003                          ERROR   .3                      ;FPP ERROR
11226                                                                         ;NO GO TO ERROR
11227 054250  012637  000004          4$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11228
11229                                 ;
11230 054254  000167  000004                    JMP     FIN14
11231                                 ;
11232 054260  104003                  TSF14:  ERROR   .3                      ;FPP ERROR
```

FLOATING POINT TESTS

```
11233                                                           ;ODD ADDRESS TRAP
11234 054262  000006                    RTT                             ;RETURN
11235                                   ;
11236 054264  000240                    FIN14:  NOP
11237                                   ;
11240                                   ;
11241                                   ;---------------------------------
11242                                   ;TEST  FDST SOP MODE 3
11243                                   ;
11244                                   ;
11245 054266                            TSFP15:
11246                                   ;
11247 054266  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11248 054272  012737  054446  000004            MOV     #TSF15,@#4              ;SETUP NEW VECTOR
11249 054300  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11250 054304  012702  000200                    MOV     #200,R2                 ;SETUP TO LOAD FPS
11251 054310  170102                            LDFPS   R2                      ;SET FD=1
11252 054312  012705  000011                    MOV     #9.,R5                  ;INIT COUNTER
11253 054316  012704  003162                    MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11254 054322  012724  177777            100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11255 054326  077503                            SOB     R5,100$                 ;ARE WE DONE
11256 054330  012737  003174  003162            MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11257 054336  012702  003162                    MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11258 054342  170432                            CLRD    @(R2)+                  ; TEST INSTRUCTION
11259 054344  170203                            STFPS   R3                      ;GET FPS
11260 054346  020227  003164                    CMP     R2,#TSTLOC+2                    ;IS R2 CORRECT
11261 054352  001401                            BEQ     1$                      ;YES GO ON
11262 054354  104003                            ERROR   +3              ;FPP ERROR
11263                                                                   ;NO GO TO ERROR
11264 054356  012702  003162            1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11265 054362  022227  003174                    CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11266 054366  001401                            BEQ     2$              ;YES GO ON
11267 054370  104003                            ERROR   +3              ;FPP ERROR
11268                                                                   ;NO GO TO ERROR
11269 054372  012701  000004            2$:     MOV     #4,R1                   ;INIT COUNTER
11270 054376  022227  177777            3$:     CMP     (R2)+,#177777           ;IS LOCATION ALL ONES
11271 054402  001401                            BEQ     4$                      ;YES GO ON
11272 054404  104003                            ERROR   +3              ;FPP ERROR
11273                                                                   ;NO GO TO ERROR
11274 054406  077105                    4$:     SOB     R1,3$                   ;ARE WE DONE
11275 054410  012701  000004                    MOV     #4,R1                   ;INIT COUNTER
11276 054414  022227  000000            5$:     CMP     (R2)+,#0                ;IS LOCATION 0
11277 054420  001401                            BEQ     6$                      ;YES GO ON
11278 054422  104003                            ERROR   +3              ;FPP ERROR
11279                                                                   ;NO GO TO ERROR
11280 054424  077105                    6$:     SOB     R1,5$                   ;ARE WE DONE
11281 054426  020327  000204                    CMP     R3,#204                 ;CHECK FPS
11282 054432  001401                            BEQ     7$                      ;OK GO ON
11283 054434  104003                            ERROR   +3              ;FPP ERROR
11284                                                                   ;NO GO TO ERROR
11285 054436  012637  000004            7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11286
11287                                   ;
11288 054442  000167  000004                    JMP     FIN15
11289                                   ;
11290 054446  104003                    TSF15:  ERROR   +3              ;FPP ERROR
11291                                                                   ;ODD ADDRESS TRAP
```

FLOATING POINT TESTS

```
11292 054450  000006                          RTT                             ;RETURN
11293                                 ;
11294 054452  000240                  FIN15:  NOP
11295                                 ;
11298                                 ;
11299                                 ;---------------------------------
11300                                 ;TEST  FDST SOP MODE 4
11301                                 ;
11302                                 ;
11303 054454                          TSFP16:
11304                                 ;
11305 054454  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11306 054460  012737  054612  000004         MOV     #TSF16,@#4              ;SETUP NEW VECTOR
11307 054466  005037  135370                 CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11308 054472  012702  000200                 MOV     #200,R2                 ;SETUP TO LOAD FPS
11309 054476  170102                         LDFPS   R2                      ;SET FD=1
11310 054500  012705  000010                 MOV     #8.,R5                  ;INIT COUNTER
11311 054504  012704  003162                 MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11312 054510  012724  177777          100$:  MOV     #177777,(R4)+           ;INIT TEST LOCATION
11313 054514  077503                         SOB     R5,100$                 ;ARE WE DONE
11314 054516  012702  003172                 MOV     #TSTLOC+10,R2                   ;SETUP POINTER TO DATA
11315 054522  170442                         CLRD    -(R2)                   ; TEST INSTRUCTION
11316 054524  170203                         STFPS   R3                      ;GET FPS
11317 054526  020227  003162                 CMP     R2,#TSTLOC                      ;IS R2 CORRECT
11318 054532  001401                         BEQ     1$                      ;YES GO ON
11319 054534  104003                         ERROR   .3              ;FPP ERROR
11320                                                                 ;NO GO TO ERROR
11321 054536  012701  000004          1$:    MOV     #4,R1                   ;INIT COUNTER
11322 054542  022227  000000          2$:    CMP     (R2)+,#0                ;IS LOCATION 0
11323 054546  001401                         BEQ     3$                      ;YES GO ON
11324 054550  104003                         ERROR   .3              ;FPP ERROR
11325                                                                 ;NO GO TO ERROR
11326 054552  077105                  3$:    SOB     R1,2$                   ;ARE WE DONE
11327 054554  012701  000004                 MOV     #4,R1                   ;INIT COUNTER
11328 054560  022227  177777          4$:    CMP     (R2)+,#177777           ;IS LOCATION UNCHANGED
11329 054564  001401                         BEQ     5$                      ;YES GO ON
11330 054566  104003                         ERROR   .3              ;FPP ERROR
11331                                                                 ;NO GO TO ERROR
11332 054570  077105                  5$:    SOB     R1,4$                   ;ARE WE DONE
11333 054572  020327  000204                 CMP     R3,#204                 ;CHECK FPS
11334 054576  001401                         BEQ     6$                      ;OK GO ON
11335 054600  104003                         ERROR   .3              ;FPP ERROR
11336                                                                 ;NO GO TO ERROR
11337 054602  012637  000004          6$:    MOV     (SP)+,@#4               ;RESTORE VECTOR
11338
11339                                 ;
11340 054606  000167  000004                  JMP     FIN16
11341                                 ;
11342 054612  104003                  TSF16:  ERROR   .3              ;FPP ERROR
11343                                                                 ;ODD ADDRESS TRAP
11344 054614  000006                          RTT                             ;RETURN
11345                                 ;
11346 054616  000240                  FIN16:  NOP
11347                                 ;
11350                                 ;
11351                                 ;---------------------------------
11352                                 ;TEST  FDST SOP MODE 5
```

```
11353                                   ;
11354                                   ;
11355 054620                            TSFP17:
11356                                   ;
11357 054620  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11358 054624  012737  054774  000004            MOV     #TSF17,@#4              ;SETUP NEW VECTOR
11359 054632  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11360 054636  012702  000200                    MOV     #200,R2                 ;SETUP TO LOAD FPS
11361 054642  170102                            LDFPS   R2                      ;SET FD=1
11362 054644  012705  000011                    MOV     #9.,R5                  ;INIT COUNTER
11363 054650  012704  003162                    MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11364 054654  012724  177777            100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11365 054660  077503                            SOB     R5,100$                 ;ARE WE DONE
11366 054662  012737  003174  003162            MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11367 054670  012702  003164                    MOV     #TSTLOC+2,R2                    ;SETUP POINTER TO DATA
11368 054674  170452                            CLRD    @-(R2)                  ; TEST INSTRUCTION
11369 054676  170203                            STFPS   R3                      ;GET FPS
11370 054700  020227  003162                    CMP     R2,#TSTLOC                      ;IS R2 CORRECT
11371 054704  001401                            BEQ     1$                      ;YES GO ON
11372 054706  104003                            ERROR   .3              ;FPP ERROR
11373                                           ;NO GO TO ERROR
11374 054710  022227  003174            1$:     CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11375 054714  001401                            BEQ     2$                      ;YES GO ON
11376 054716  104003                            ERROR   .3              ;FPP ERROR
11377                                           ;NO GO TO ERROR
11378 054720  012701  000004            2$:     MOV     #4,R1                   ;INIT COUNTER
11379 054724  022227  177777            3$:     CMP     (R2)+,#177777           ;IS LOCATION ALL ONES
11380 054730  001401                            BEQ     4$                      ;YES GO ON
11381 054732  104003                            ERROR   .3              ;FPP ERROR
11382                                           ;NO GO TO ERROR
11383 054734  077105                    4$:     SOB     R1,3$                   ;ARE WE DONE
11384 054736  012701  000004                    MOV     #4,R1                   ;INIT COUNTER
11385 054742  022227  000000            5$:     CMP     (R2)+,#0                ;IS LOCATION 0
11386 054746  001401                            BEQ     6$                      ;YES GO ON
11387 054750  104003                            ERROR   .3              ;FPP ERROR
11388                                           ;NO GO TO ERROR
11389 054752  077105                    6$:     SOB     R1,5$                   ;ARE WE DONE
11390 054754  020327  000204                    CMP     R3,#204                 ;CHECK FPS
11391 054760  001401                            BEQ     7$                      ;OK GO ON
11392 054762  104003                            ERROR   .3              ;FPP ERROR
11393                                           ;NO GO TO ERROR
11394 054764  012637  000004            7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11395
11396                                   ;
11397 054770  000167  000004                    JMP     FIN17
11398                                   ;
11399 054774  104003                    TSF17:  ERROR   .3              ;FPP ERROR
11400                                           ;ODD ADDRESS TRAP
11401 054776  000006                            RTT                             ;RETURN
11402                                   ;
11403 055000  000240                    FIN17:  NOP
11404                                   ;
11407                                   ;
11408                                   ;---------------------------------
11409                                   ;TEST  FDST SOP MODE 6
11410                                   ;
11411                                   ;
```

FLOATING POINT TESTS

```
11412 055002                                TSFP20:
11413                                       ;
11414 055002  005037  135370                        CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11415 055006  013746  000004                        MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11416 055012  012737  055146  000004                MOV     #TSF20,@#4              ;SETUP NEW VECTOR
11417 055020  012702  000200                        MOV     #200,R2                 ;SETUP TO LOAD FPS
11418 055024  170102                                LDFPS   R2                      ;SET FD=1
11419 055026  012705  000010                        MOV     #8.,R5                  ;INIT COUNTER
11420 055032  012704  003162                        MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11421 055036  012724  177777                100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11422 055042  077503                                SOB     R5,100$                 ;ARE WE DONE
11423 055044  012702  003163                        MOV     #TSTLOC+1,R2                    ;SETUP POINTER TO DATA
11424 055050  170462  000007                        CLRD    7(R2)                   ; TEST INSTRUCTION
11425 055054  170203                                STFPS   R3                      ;GET FPS
11426 055056  020227  003163                        CMP     R2,#TSTLOC+1                    ;IS R2 CORRECT
11427 055062  001401                                BEQ     1$                      ;YES GO ON
11428 055064  104003                                ERROR   +3              ;FPP ERROR
11429                                               ;NO GO TO ERROR
11430 055066  012702  003162                1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11431 055072  012701  000004                        MOV     #4,R1                   ;INIT COUNTER
11432 055076  022227  177777                2$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11433 055102  001401                                BEQ     3$                      ;YES GO ON
11434 055104  104003                                ERROR   +3              ;FPP ERROR
11435                                               ;NO GO TO ERROR
11436 055106  077105                        3$:     SOB     R1,2$                   ;ARE WE DONE
11437 055110  012701  000004                        MOV     #4,R1                   ;INIT COUNTER
11438 055114  022227  000000                4$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11439 055120  001401                                BEQ     5$                      ;YES GO ON
11440 055122  104003                                ERROR   +3              ;FPP ERROR
11441                                               ;NO GO TO ERROR
11442 055124  077105                        5$:     SOB     R1,4$                   ;ARE WE DONE
11443 055126  020327  000204                        CMP     R3,#204                 ;IS FPS CORRECT
11444 055132  001401                                BEQ     6$                      ;YES GO ON
11445 055134  104003                                ERROR   +3              ;FPP ERROR
11446                                               ;NO GO TO ERROR
11447 055136  012637  000004                6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11448
11449                                       ;
11450 055142  000167  000004                        JMP     FIN20
11451                                       ;
11452 055146  104003                        TSF20:  ERROR   +3              ;FPP ERROR
11453                                               ;ODD ADDRESS TRAP
11454 055150  000006                                RTT                             ;RETURN
11455                                       ;
11456 055152  000240                        FIN20:  NOP
11457                                       ;
11460                                       ;
11461                                       ;-----------------------------------
11462                                       ;TEST  FDST SOP MODE 7
11463                                       ;
11464                                       ;
11465 055154                                TSFP21:
11466                                       ;
11467 055154  005037  135370                        CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11468 055160  013746  000004                        MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11469 055164  012737  055336  000004                MOV     #TSF21,@#4              ;SETUP NEW VECTOR
11470 055172  012702  000200                        MOV     #200,R2                 ;SETUP TO LOAD FPS
```

FLOATING POINT TESTS

```
11471 055176  170102                        LDFPS   R2                      ;SET FD=1
11472 055200  012705  000010                MOV     #8.,R5                  ;INIT COUNTER
11473 055204  012704  003162                MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11474 055210  012724  177777        100$:   MOV     #177777,(R4)+   ;INIT TEST LOCATION
11475 055214  077503                        SOB     R5,100$                 ;ARE WE DONE
11476 055216  012737  003162  003172        MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11477 055224  012702  003165                MOV     #TSTLOC+3,R2                    ;SETUP POINTER TO DATA
11478 055230  170472  000005                CLRD    @5(R2)                  ; TEST INSTRUCTION
11479 055234  170203                        STFPS   R3                      ;GET FPS
11480 055236  020227  003165                CMP     R2,#TSTLOC+3                    ;IS R2 CORRECT
11481 055242  001401                        BEQ     1$                      ;YES GO ON
11482 055244  104003                        ERROR   +3              ;FPP ERROR
11483                                               ;NO GO TO ERROR
11484 055246  012702  003162          1$:   MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11485 055252  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11486 055256  022227  000000          2$:   CMP     (R2)+,#0                ;IS DATA CORRECT
11487 055262  001401                        BEQ     3$                      ;YES GO ON
11488 055264  104003                        ERROR   +3              ;FPP ERROR
11489                                               ;NO GO TO ERROR
11490 055266  077105                  3$:   SOB     R1,2$                   ;ARE WE DONE
11491 055270  022227  003162                CMP     (R2)+,#TSTLOC           ;IS DATA CORRECT
11492 055274  001401                        BEQ     4$                      ;YES GO ON
11493 055276  104003                        ERROR   +3              ;FPP ERROR
11494                                               ;NO GO TO ERROR
11495 055300  012701  000003          4$:   MOV     #3,R1                   ;INIT COUNTER
11496 055304  022227  177777          5$:   CMP     (R2)+,#177777           ;IS DATA CORRECT
11497 055310  001401                        BEQ     6$                      ;YES GO ON
11498 055312  104003                        ERROR   +3              ;FPP ERROR
11499                                               ;NO GO TO ERROR
11500 055314  077105                  6$:   SOB     R1,5$                   ;ARE WE DONE
11501 055316  020327  000204                CMP     R3,#204                 ;CHECK FPS
11502 055322  001401                        BEQ     7$                      ;OK GO ON
11503 055324  104003                        ERROR   +3              ;FPP ERROR
11504                                               ;NO GO TO ERROR
11505 055326  012637  000004          7$:   MOV     (SP)+,@#4               ;RESTORE VECTOR
11506
11507                                       ;
11508 055332  000167  000004                JMP     FIN21
11509                                       ;
11510 055336  104003                  TSF21:  ERROR   +3              ;FPP ERROR
11511                                               ;ODD ADDRESS TRAP
11512 055340  000006                        RTT                             ;RETURN
11513                                       ;
11514 055342  000240                  FIN21:  NOP
11515                                       ;
11518                                       ;
11519                                       ;---------------------------------
11520                                       ;TEST  FDST SOP MODE 3 GR7
11521                                       ;
11522                                       ;
11523 055344                          TSFP22:
11524                                       ;
11525 055344  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11526 055350  013746  000004                MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
11527 055354  012737  055502  000004        MOV     #TSF22,@#4              ;SETUP NEW VECTOR
11528 055362  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11529 055366  170102                        LDFPS   R2                      ;SET FD=1
```

FLOATING POINT TESTS

```
11530 055370  012705  000010                MOV     #8.,R5                  ;INIT COUNTER
11531 055374  012704  003162                MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11532 055400  012724  177777        100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11533 055404  077503                        SOB     R5,100$                 ;ARE WE DONE
11534 055406  012737  003172 003162         MOV     #TSTLOC+10,@#TSTLOC             ;INIT TEST LOCATION
11535 055414  170437  003162                CLRD    @#TSTLOC                        ; TEST INSTRUCTION
11536 055420  170203                        STFPS   R3                      ;GET FPS
11537 055422  012702  003162                MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11538 055426  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11539 055432  022227  000000        1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11540 055436  001401                        BEQ     2$                      ;YES GO ON
11541 055440  104003                        ERROR.  +3              ;FPP ERROR
11542                                                               ;NO GO TO ERROR
11543 055442  077105                2$:     SOB     R1,1$                   ;ARE WE DONE
11544 055444  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11545 055450  022227  177777        3$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11546 055454  001401                        BEQ     4$                      ;YES GO ON
11547 055456  104003                        ERROR   +3              ;FPP ERROR
11548                                                               ;NO GO TO ERROR
11549 055460  077105                4$:     SOB     R1,3$                   ;ARE WE DONE
11550 055462  020327  000204                CMP     R3,#204                 ;CHECK FPS
11551 055466  001401                        BEQ     5$                      ;OK GO ON
11552 055470  104003                        ERROR   +3              ;FPP ERROR
11553                                                               ;NO GO TO ERROR
11554 055472  012637  000004        5$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11555
11556                               ;
11557 055476  000167  000004                JMP     FIN22
11558                               ;
11559 055502  104003                TSF22:  ERROR   +3              ;FPP ERROR
11560                                                               ;ODD ADDRESS TRAP
11561 055504  000006                        RTT                             ;RETURN
11562                               ;
11563 055506  000240                FIN22:  NOP
11564                               ;
11567                               ;
11568                               ;---------------------------------
11569                               ;TEST  FDST SOP MODE 6 GR7
11570                               ;
11571                               ;
11572 055510                        TSFP23:
11573                               ;
11574 055510  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11575 055514  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11576 055520  012737  055622 000004         MOV     #TSF23,@#4              ;SETUP NEW VECTOR
11577 055526  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11578 055532  170102                        LDFPS   R2                      ;SET FD=1
11579 055534  012705  000004                MOV     #4,R5                   ;INIT COUNTER
11580 055540  012704  003162                MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11581 055544  012724  177777        100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11582 055550  077503                        SOB     R5,100$                 ;ARE WE DONE
11583 055552  170467  125404                CLRD    TSTLOC                  ; TEST INSTRUCTION
11584 055556  170203                        STFPS   R3                      ;GET FPS
11585 055560  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11586 055564  012702  003162                MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11587 055570  022227  000000        1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11588 055574  001401                        BEQ     2$                      ;YES GO ON
```

FLOATING POINT TESTS

```
11589 055576  104003                        ERROR   .3              ;FPP ERROR
11590                                                               ;NO GO TO ERROR
11591 055600  077105                  2$:   SOB     R1,1$                   ;ARE WE DONE
11592 055602  020327  000204                CMP     R3,#204                 ;CHECK FPS
11593 055606  001401                        BEQ     3$                      ;OK GO ON
11594 055610  104003                        ERROR   .3              ;FPP ERROR
11595                                                               ;NO GO TO ERROR
11596 055612  012637  000004          3$:   MOV     (SP)+,@#4               ;RESTORE VECTOR
11597
11598                                 ;
11599 055616  000167  000004                JMP     FIN23
11600                                 ;
11601 055622  104003                  TSF23:ERROR   .3              ;FPP ERROR
11602                                                               ;ODD ADDRESS TRAP
11603 055624  000006                        RTT                             ;
11604 055626  000240                  FIN23:NOP
11605                                 ;
11608                                 ;
11609                                 ;---------------------------------
11610                                 ;TEST  FDST SOP MODE 7 GR7
11611                                 ;
11612                                 ;
11613 055630                          TSFP24:
11614                                 ;
11615 055630  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11616 055634  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11617 055640  012737  055776  000004        MOV     #TSF24,@#4              ;SETUP NEW VECTOR
11618 055646  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11619 055652  170102                        LDFPS   R2                      ;SET FD=1
11620 055654  012705  000010                MOV     #8.,R5          ;INIT COUNTER
11621 055660  012704  003162                MOV     #TSTLOC,R4              ;SETUP TEST LOCATION POINTER
11622 055664  012724  177777          100$: MOV     #177777,(R4)+           ;INIT TEST LOCATION
11623 055670  077503                        SOB     R5,100$                 ;ARE WE DONE
11624 055672  012737  003162  003172        MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11625 055700  170477  125266                CLRD    @TSTLOC+10                      ; TEST INSTRUCTION
11626 055704  170203                        STFPS   R3                      ;GET FPS
11627 055706  012702  003162                MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11628 055712  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11629 055716  022227  000000          1$:   CMP     (R2)+,#0                ;IS DATA CORRECT
11630 055722  001401                        BEQ     2$                      ;YES GO ON
11631 055724  104003                        ERROR   .3              ;FPP ERROR
11632                                                               ;NO GO TO ERROR
11633 055726  077105                  2$:   SOB     R1,1$                   ;ARE WE DONE
11634 055730  022227  003162                CMP     (R2)+,#TSTLOC           ;IS DATA CORRECT
11635 055734  001401                        BEQ     3$                      ;YES GO ON
11636 055736  104003                        ERROR   .3              ;FPP ERROR
11637                                                               ;NO GO TO ERROR
11638 055740  012701  000003          3$:   MOV     #3,R1                   ;INIT COUNTER
11639 055744  022227  177777          4$:   CMP     (R2)+,#177777           ;IS DATA CORRECT
11640 055750  001401                        BEQ     5$                      ;YES GO ON
11641 055752  104003                        ERROR   .3              ;FPP ERROR
11642                                                               ;NO GO TO ERROR
11643 055754  077105                  5$:   SOB     R1,4$                   ;ARE WE DONE
11644 055756  020327  000204                CMP     R3,#204                 ;CHECK FPS
11645 055762  001401                        BEQ     6$                      ;OK GO ON
11646 055764  104003                        ERROR   .3              ;FPP ERROR
11647                                                               ;NO GO TO ERROR
```

FLOATING POINT TESTS

```
11648 055766  012637  000004          6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11649
11650                                 ;
11651 055772  000167  000004                  JMP     FIN24
11652                                 ;
11653 055776  104003                  TSF24:  ERROR   +3              ;FPP ERROR
11654                                                                 ;ODD ADDRESS TRAP
11655 056000  000006                          RTT                             ;
11656                                 ;
11657 056002  000240                  FIN24:  NOP
11658                                 ;
11661                                 ;
11662                                 ;---------------------------------
11663                                 ;TEST  CLRF
11664                                 ;
11665                                 ;
11666 056004                          TSFP25:
11667                                 ;
11668 056004  005037  135370                  CLR     @#@#TRPFLG              ;CLEAR TRAP FLAG
11669 056010  005002                          CLR     R2                      ;SETUP TO LOAD FPS
11670 056012  170102                          LDFPS   R2                      ;SET FD=0
11671 056014  012705  000004                  MOV     #4,R5                   ;INIT COUNTER
11672 056020  012704  003162                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11673 056024  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11674 056030  077503                          SOB     R5,100$                 ;ARE WE DONE
11675 056032  012702  003162                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11676 056036  170422                          CLRF    (R2)+                   ; TEST INSTRUCTION
11677 056040  170203                          STFPS   R3                      ;GET FPS
11678 056042  020227  003166                  CMP     R2,#TSTLOC+4                    ;IS R2 CORRECT
11679 056046  001401                          BEQ     1$                      ;YES GO ON
11680 056050  104003                          ERROR   +3              ;FPP ERROR
11681                                                                 ;NO GO TO ERROR
11682 056052  012702  003162          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11683 056056  012701  000002                  MOV     #2,R1                   ;INIT COUNTER
11684 056062  022227  000000          2$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11685 056066  001401                          BEQ     3$                      ;YES GO ON
11686 056070  104003                          ERROR   +3              ;FPP ERROR
11687                                                                 ;NO GO TO ERROR
11688 056072  077105                  3$:     SOB     R1,2$                   ;ARE WE DONE
11689 056074  012701  000002                  MOV     #2,R1                   ;INIT COUNTER
11690 056100  022227  177777          4$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11691 056104  001401                          BEQ     5$                      ;YES GO ON
11692 056106  104003                          ERROR   +3              ;FPP ERROR
11693                                                                 ;NO GO TO ERROR
11694 056110  077105                  5$:     SOB     R1,4$                   ;ARE WE DONE
11695 056112  020327  000004                  CMP     R3,#4                   ;CHECK FPS
11696 056116  001401                          BEQ     6$                      ;OK GO ON
11697 056120  104003                          ERROR   +3              ;FPP ERROR
11698                                                                 ;NO GO TO ERROR
11699 056122                          6$:
11700                                 ;
11703                                 ;
11704                                 ;---------------------------------
11705                                 ;TEST  TSTF AND TSTD
11706                                 ;
11707                                 ;
11708 056122                          TSFP26:
```

FLOATING POINT TESTS

```
11709                                   ;
11710 056122  005037  135370                      CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11711 056126  005004                              CLR     R4                      ;SETUP TO LOAD FPS
11712 056130  170104                              LDFPS   R4                      ;SET FD=0
11713 056132  170567  000244                      TSTF    TS26D0                  ; TEST INSTRUCTION
11714 056136  170203                              STFPS   R3                      ;GET FPS
11715 056140  020327  000004                      CMP     R3,#4                   ;CHECK FPS
11716 056144  001401                              BEQ     1$                      ;OK GO ON
11717 056146  104003                              ERROR   .3              ;FPP ERROR
11718                                           ;NO GO TO ERROR
11719 056150  012704  056402            1$:       MOV     #TS26D0,R4              ;SETUP POINTERS TO DATA
11720 056154  012702  056432                      MOV     #TS26D3,R2              ;
11721 056160  004767  000200                      JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11722 056164  170537  056412                      TSTF    @#TS26D1                ; TEST INSTRUCTION
11723 056170  170203                              STFPS   R3                      ;GET FPS
11724 056172  020327  000010                      CMP     R3,#10                  ;CHECK FPS
11725 056176  001401                              BEQ     2$                      ;OK GO ON
11726 056200  104003                              ERROR   .3              ;FPP ERROR
11727                                           ;NO GO TO ERROR
11728 056202  012704  056412            2$:       MOV     #TS26D1,R4              ;SETUP POINTERS TO DATA
11729 056206  012702  056442                      MOV     #TS26D4,R2              ;
11730 056212  004767  000146                      JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11731 056216  170567  000200                      TSTF    TS26D2                  ; TEST INSTRUCTION
11732 056222  170203                              STFPS   R3                      ;GET FPS
11733 056224  020327  000000                      CMP     R3,#0                   ;CHECK FPS
11734 056230  001401                              BEQ     3$                      ;OK GO ON
11735 056232  104003                              ERROR   .3              ;FPP ERROR
11736                                           ;NO GO TO ERROR
11737 056234  012704  056422            3$:       MOV     #TS26D2,R4              ;SETUP POINTERS TO DATA
11738 056240  012702  056452                      MOV     #TS26D5,R2              ;
11739 056244  004767  000114                      JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11740 056250  012704  000200                      MOV     #200,R4                 ;SETUP TO LOAD FPS
11741 056254  170104                              LDFPS   R4                      ;SET FD=1
11742 056256  170537  056402                      TSTD    @#TS26D0                ; TEST INSTRUCTION
11743 056262  170203                              STFPS   R3                      ;GET FPS
11744 056264  020327  000204                      CMP     R3,#204                 ;CHECK FPS
11745 056270  001401                              BEQ     4$                      ;OK GO ON
11746 056272  104003                              ERROR   .3              ;FPP ERROR
11747                                           ;NO GO TO ERROR
11748 056274  012704  056402            4$:       MOV     #TS26D0,R4              ;SETUP POINTERS TO DATA
11749 056300  012702  056432                      MOV     #TS26D3,R2              ;
11750 056304  004767  000054                      JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11751 056310  170567  000076                      TSTD    TS26D1                  ; TEST INSTRUCTION
11752 056314  170203                              STFPS   R3                      ;GET FPS
11753 056316  020327  000210                      CMP     R3,#210                 ;CHECK FPS
11754 056322  001401                              BEQ     5$                      ;OK GO ON
11755 056324  104003                              ERROR   .3              ;FPP ERROR
11756                                           ;NO GO TO ERROR
11757 056326  012704  056412            5$:       MOV     #TS26D1,R4              ;SETUP POINTERS TO DATA
11758 056332  012702  056442                      MOV     #TS26D4,R2              ;
11759 056336  004767  000022                      JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11760 056342  170567  000054                      TSTD    TS26D2                  ; TEST INSTRUCTION
11761 056346  170203                              STFPS   R3                      ;GET FPS
11762 056350  020327  000200                      CMP     R3,#200                 ;CHECK FPS
11763 056354  001401                              BEQ     6$                      ;OK GO ON
11764 056356  104003                              ERROR   .3              ;FPP ERROR
11765                                           ;NO GO TO ERROR
```

FLOATING POINT TESTS

```
11766 056360                          6$:
11767                                 ;
11768 056360  000167  000076                  JMP     FIN26
11769                                 ;
11770 056364  012701  000004          CHEC26: MOV     #4,R1                   ;INIT COUNTER
11771 056370  022422                  1$:     CMP     (R4)+,(R2)+             ;IS DATA CORRECT
11772 056372  001401                          BEQ     2$                      ;YES GO ON
11773 056374  104003                          ERROR   .3              ;FPP ERROR
11774                                                                 ;NO GO TO ERROR
11775 056376  077104                  2$:     SOB     R1,1$                   ;ARE WE DONE
11776 056400  000207                          RTS     PC                      ;RETURN
11777                                 ;
11778 056402  000177                  TS26D0: .WORD   177
11779 056404  177777                          .WORD   177777
11780 056406  177777                          .WORD   177777
11781 056410  177777                          .WORD   177777
11782 056412  177777                  TS26D1: .WORD   177777
11783 056414  000000                          .WORD   0
11784 056416  000000                          .WORD   0
11785 056420  000000                          .WORD   0
11786 056422  077777                  TS26D2: .WORD   77777
11787 056424  000000                          .WORD   0
11788 056426  000000                          .WORD   0
11789 056430  000000                          .WORD   0
11790 056432  0C0177                  TS26D3: .WORD   177
11791 056434  177777                          .WORD   177777
11792 056436  177777                          .WORD   177777
11793 056440  177777                          .WORD   177777
11794 056442  177777                  TS26D4: .WORD   177777
11795 056444  000000                          .WORD   0
11796 056446  000000                          .WORD   0
11797 056450  000000                          .WORD   0
11798 056452  077777                  TS26D5: .WORD   77777
11799 056454  000000                          .WORD   0
11800 056456  000000                          .WORD   0
11801 056460  000000                          .WORD   0
11802 056462  000240                  FIN26:  NOP
11803                                 ;
11806                                 ;
11807                                 ;-----------------------------------
11808                                 ;TEST  ABSF
11809                                 ;
11810                                 ;
11811 056464                          TSFP27:
11812                                 ;
11813 056464  005037  135370                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11814 056470  005005                          CLR     R5                      ;SETUP TO LOAD FPS
11815 056472  170105                          LDFPS   R5                      ;SET FD=0
11816 056474  012701  000014                  MOV     #12.,R1                 ;INIT COUNTER
11817 056500  012704  003162                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11818 056504  012703  056710                  MOV     #TS27D0,R3              ;SETUP POINTER TO TEST VALUE
11819 056510  012324                  100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11820 056512  077102                          SOB     R1,100$                 ;ARE WE DONE
11821 056514  012705  003162                  MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11822 056520  170615                          ABSF    (R5)                    ; TEST INSTRUCTION
11823 056522  170203                          STFPS   R3                      ;GET FPS
11824 056524  020527  003162                  CMP     R5,#TSTLOC                      ;IS R5 CORRECT
```

FLOATING POINT TESTS

```
11825 056530  001401                  BEQ     1$                ;YES GO ON
11826 056532  104003                  ERROR   .3              ;FPP ERROR
11827                                                         ;NO GO TO ERROR
11828 056534  012702  056740    1$:   MOV     @TS27D3,R2        ;SETUP POINTER TO DATA
11829 056540  004767  000126          JSR     PC,CHEC27         ;CHECK IF DATA IS CORRECT
11830 056544  020327  000000          CMP     R3,#0             ;CHECK FPS
11831 056550  001401                  BEQ     2$                ;OK GO ON
11832 056552  104003                  ERROR   .3              ;FPP ERROR
11833                                                         ;NO GO TO ERROR
11834 056554  012705  003172    2$:   MOV     @TSTLOC+10,R5                 ;SETUP POINTER TO DATA
11835 056560  170625                  ABSF    (R5)+             ; TEST INSTRUCTION
11836 056562  170203                  STFPS   R3                ;GET FPS
11837 056564  020527  003176          CMP     R5,@TSTLOC+14               ;IS R5 CORRECT
11838 056570  001401                  BEQ     3$                ;YES GO ON
11839 056572  104003                  ERROR   .3              ;FPP ERROR
11840                                                         ;NO GO TO ERROR
11841 056574  012705  003172    3$:   MOV     @TSTLOC+10,R5                 ;SETUP POINTER TO DATA
11842 056600  012702  056750          MOV     @TS27D4,R2        ;
11843 056604  004767  000062          JSR     PC,CHEC27         ;CHECK IF DATA IS CORRECT
11844 056610  020327  000000          CMP     R3,#0             ;CHECK FPS
11845 056614  001401                  BEQ     4$                ;OK GO ON
11846 056616  104003                  ERROR   .3              ;FPP ERROR
11847                                                         ;NO GO TO ERROR
11848 056620  012705  003162    4$:   MOV     @TSTLOC,R5                    ;SETUP POINTER TO DATA
11849 056624  170665  000020          ABSF    20(R5)            ; TEST INSTRUCTION
11850 056630  170203                  STFPS   R3                ;GET FPS
11851 056632  020527  003162          CMP     R5,@TSTLOC                  ;IS R5 CORRECT
11852 056636  001401                  BEQ     5$                ;YES GO ON
11853 056640  104003                  ERROR   .3              ;FPP ERROR
11854                                                         ;NO GO TO ERROR
11855 056642  012705  003202    5$:   MOV     @TSTLOC+20,R5                 ;SETUP POINTERS TO DATA
11856 056646  012702  056760          MOV     @TS27D5,R2        ;
11857 056652  004767  000014          JSR     PC,CHEC27         ;CHECK IF DATA IS CORRECT
11858 056656  020327  000004          CMP     R3,#4             ;CHECK FPS
11859 056662  001401                  BEQ     6$                ;OK GO ON
11860 056664  104003                  ERROR   .3              ;FPP ERROR
11861                                                         ;NO GO TO ERROR
11862 056666                    6$:
11863                           ;
11864 056666  000167  000076          JMP     FIN27
11865                           ;
11866 056672  012701  000004  CHEC27: MOV     #4,R1             ;INIT COUNTER
11867 056676  022522          1$:   CMP     (R5)+,(R2)+       ;IS DATA CORRECT
11868 056700  001401                  BEQ     2$                ;YES GO ON
11869 056702  104003                  ERROR   .3              ;FPP ERROR
11870                                                         ;NO GO TO ERROR
11871 056704  077104          2$:   SOB     R1,1$             ;ARE WE DONE
11872 056706  000207                  RTS     PC                ;RETURN
11873                           ;
11874 056710  177777  TS27D0: .WORD   177777
11875 056712  177777          .WORD   177777
11876 056714  177777          .WORD   177777
11877 056716  177777          .WORD   177777
11878 056720  000377  TS27D1: .WORD   377
11879 056722  175436          .WORD   175436
11880 056724  136477          .WORD   136477
11881 056726  000001          .WORD   1
```

FLOATING POINT TESTS

```
11882 056730  000177                TS27D2: .WORD   177
11883 056732  175436                        .WORD   175436
11884 056734  136477                        .WORD   136477
11885 056736  000001                        .WORD   1
11886 056740  077777                TS27D3: .WORD   77777
11887 056742  177777                        .WORD   177777
11888 056744  177777                        .WORD   177777
11889 056746  177777                        .WORD   177777
11890 056750  000377                TS27D4: .WORD   377
11891 056752  175436                        .WORD   175436
11892 056754  136477                        .WORD   136477
11893 056756  000001                        .WORD   1
11894 056760  000000                TS27D5: .WORD   0
11895 056762  000000                        .WORD   0
11896 056764  136477                        .WORD   136477
11897 056766  000001                        .WORD   1
11898 056770  000240                FIN27:  NOP
11899                               ;
11902                               ;
11903                               ;--------------------------------
11904                               ;TEST  ABSD
11905                               ;
11906                               ;
11907 056772                        TSFP30:
11908                               ;
11909 056772  005037  135370                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11910 056776  012705  000200                MOV     #200,R5                 ;SETUP TO LOAD FPS
11911 057002  170105                        LDFPS   R5                      ;SET FD=1
11912 057004  012701  000014                MOV     #12.,R1                 ;INIT COUNTER
11913 057010  012704  003162                MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11914 057014  012703  057220                MOV     #TS30D0,R3              ;SETUP POINTER TO TEST VALUE
11915 057020  012324                100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11916 057022  077102                        SOB     R1,100$                 ;ARE WE DONE
11917 057024  012705  003162                MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11918 057030  170615                        ABSD    (R5)                    ; TEST INSTRUCTION
11919 057032  170203                        STFPS   R3                      ;GET FPS
11920 057034  020527  003162                CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11921 057040  001401                        BEQ     1$                      ;YES GO ON
11922 057042  104003                        ERROR   .3              ;FPP ERROR
11923                                                               ;NO GO TO ERROR
11924 057044  012702  057250          1$:   MOV     #TS30D3,R2              ;SETUP POINTER TO DATA
11925 057050  004767  000126                JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11926 057054  020327  000200                CMP     R3,#200                 ;CHECK FPS
11927 057060  001401                        BEQ     2$                      ;OK GO ON
11928 057062  104003                        ERROR   .3              ;FPP ERROR
11929                                                               ;NO GO TO ERROR
11930 057064  012705  003172          2$:   MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
11931 057070  170625                        ABSD    (R5)+                   ; TEST INSTRUCTION
11932 057072  170203                        STFPS   R3                      ;GET FPS
11933 057074  020527  003202                CMP     R5,#TSTLOC+20                   ;IS R5 CORRECT
11934 057100  001401                        BEQ     3$                      ;YES GO ON
11935 057102  104003                        ERROR   .3              ;FPP ERROR
11936                                                               ;NO GO TO ERROR
11937 057104  012705  003172          3$:   MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
11938 057110  012702  057260                MOV     #TS30D4,R2              ;
11939 057114  004767  000062                JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11940 057120  020327  000200                CMP     R3,#200                 ;CHECK FPS
```

FLOATING POINT TESTS

```
11941 057124  001401                    BEQ     4$                      ;OK GO ON
11942 057126  104003                    ERROR   .3                    ;FPP ERROR
11943                                                                   ;NO GO TO ERROR
11944 057130  012705  003162      4$:     MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11945 057134  170665  000020              ABSD    20(R5)                  ; TEST INSTRUCTION
11946 057140  170203                      STFPS   R3                      ;GET FPS
11947 057142  020527  003162              CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11948 057146  001401                      BEQ     5$                      ;YES GO ON
11949 057150  104003                      ERROR   .3                    ;FPP ERROR
11950                                                                   ;NO GO TO ERROR
11951 057152  012705  003202      5$:     MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
11952 057156  012702  057270              MOV     #TS30D5,R2              ;
11953 057162  004767  000014              JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
11954 057166  020327  000204              CMP     R3,#204                 ;CHECK FPS
11955 057172  001401                      BEQ     6$                      ;OK GO ON
11956 057174  104003                      ERROR   .3                    ;FPP ERROR
11957                                                                   ;NO GO TO ERROR
11958 057176                      6$:
11959                             ;
11960 057176  000167  000076              JMP     FIN30
11961                             ;
11962 057202  012701  000004      CHEC30: MOV     #4,R1                   ;INIT COUNTER
11963 057206  022522              1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
11964 057210  001401                      BEQ     2$                      ;YES GO ON
11965 057212  104003                      ERROR   .3                    ;FPP ERROR
11966                                                                   ;NO GO TO ERROR
11967 057214  077104              2$:     SOB     R1,1$                   ;ARE WE DONE
11968 057216  000207                      RTS     PC                      ;RETURN
11969                             ;
11970 057220  177777              TS30D0: .WORD   177777
11971 057222  177777                      .WORD   177777
11972 057224  177777                      .WORD   177777
11973 057226  177777                      .WORD   177777
11974 057230  000377              TS30D1: .WORD   377
11975 057232  175436                      .WORD   175436
11976 057234  136477                      .WORD   136477
11977 057236  000001                      .WORD   1
11978 057240  000177              TS30D2: .WORD   177
11979 057242  175436                      .WORD   175436
11980 057244  136477                      .WORD   136477
11981 057246  000001                      .WORD   1
11982 057250  077777              TS30D3: .WORD   77777
11983 057252  177777                      .WORD   177777
11984 057254  177777                      .WORD   177777
11985 057256  177777                      .WORD   177777
11986 057260  000377              TS30D4: .WORD   377
11987 057262  175436                      .WORD   175436
11988 057264  136477                      .WORD   136477
11989 057266  000001                      .WORD   1
11990 057270  000000              TS30D5: .WORD   0
11991 057272  000000                      .WORD   0
11992 057274  000000                      .WORD   0
11993 057276  000000                      .WORD   0
11994 057300  000240              FIN30:  NOP
11995                             ;
11998                             ;
11999                             ;---------------------------------
```

FLOATING POINT TESTS

```
12000                                   ;TEST  FDST SOP MODE 2 GR7
12001                                   ;
12002                                   ;
12003 057302                            TSFP31:
12004                                   ;
12005 057302  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12006 057306  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
12007 057312  012737  057414  000004            MOV     #TSF31,@#4              ;SETUP NEW VECTOR
12008 057320  012702  000200                    MOV     #200,R2                 ;SETUP TO LOAD FPS
12009 057324  170102                            LDFPS   R2                      ;SET FD=1
12010 057326  170527  000005            TSD31:  TSTD    #5                      ; TEST INSTRUCTION
12011 057332  000240                            NOP
12012 057334  000240                            NOP
12013 057336  000240                            NOP
12014 057340  170203                            STFPS   R3                      ;GET FPS
12015 057342  020327  000204                    CMP     R3,#204                 ;CHECK FPS
12016 057346  001401                            BEQ     1$                      ;OK GO ON
12017 057350  104003                            ERROR   +3                      ;FPP ERROR
12018                                                                           ;NO GO TO ERROR
12019 057352  012702  057330            1$:     MOV     #TSD31+2,R2             ;SETUP POINTER TO DATA
12020 057356  022227  000005                    CMP     (R2)+,#5                ;IS DATA CORRECT
12021 057362  001401                            BEQ     2$                      ;YES GO ON
12022 057364  104003                            ERROR   +3                      ;FPP ERROR
12023                                                                           ;NO GO TO ERROR
12024 057366  012701  000003            2$:     MOV     #3,R1                   ;INIT COUNTER
12025 057372  022227  000240            3$:     CMP     (R2)+,#240              ;IS DATA CORRECT
12026 057376  001401                            BEQ     4$                      ;YES GO ON
12027 057400  104003                            ERROR   +3                      ;FPP ERROR
12028                                                                           ;NO GO TO ERROR
12029 057402  077105                    4$:     SOB     R1,3$                   ;ARE WE DONE
12030 057404  012637  000004                    MOV     (SP)+,@#4               ;RESTORE VECTOR
12031
12032                                   ;
12033 057410  000167  000004                    JMP     FIN31
12034                                   ;
12035 057414  104003                    TSF31:  ERROR   +3                      ;FPP ERROR
12036                                                                           ;ODD ADDRESS TRAP
12037 057416  000006                            RTT                             ;RETURN
12038                                   ;
12039 057420  000240                    FIN31:  NOP
12040                                   ;
12043                                   ;
12044                                   ;---------------------------------
12045                                   ;TEST  NEGF
12046                                   ;
12047                                   ;
12048 057422                            TSFP32:
12049                                   ;
12050 057422  005037  135370                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12051 057426  005005                            CLR     R5                      ;SETUP TO LOAD FPS
12052 057430  170105                            LDFPS   R5                      ;SET FD=0
12053 057432  012701  000014                    MOV     #12.,R1                 ;INIT COUNTER
12054 057436  012704  003162                    MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
12055 057442  012703  057612                    MOV     #TS32D0,R3              ;SETUP POINTER TO TEST VALUE
12056 057446  012324                    100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
12057 057450  077102                            SOB     R1,100$                 ;ARE WE DONE
12058 057452  170767  123504                    NEGF    TSTLOC                  ; TEST INSTRUCTION
```

FLOATING POINT TESTS

```
12059 057456  170203                    STFPS   R3                      ;GET FPS
12060 057460  012705  003162            MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
12061 057464  012702  057642            MOV     #TS32D3,R2              ;
12062 057470  004767  000100            JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12063 057474  020327  000000            CMP     R3,#0                   ;CHECK FPS
12064 057500  001401                    BEQ     1$                      ;YES GO ON
12065 057502  104003                    ERROR   +3              ;FPP ERROR
12066                                   ;NO GO TO ERROR
12067 057504  170767  123462    1$:     NEGF    TSTLOC+10                       ; TEST INSTRUCTION
12068 057510  170203                    STFPS   R3                      ;GET FPS
12069 057512  012705  003172            MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
12070 057516  012702  057652            MOV     #TS32D4,R2              ;
12071 057522  004767  000046            JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12072 057526  020327  000010            CMP     R3,#10                  ;CHECK FPS
12073 057532  001401                    BEQ     2$                      ;OK GO ON
12074 057534  104003                    ERROR   +3              ;FPP ERROR
12075                                   ;NO GO TO ERROR
12076 057536  170767  123440    2$:     NEGF    TSTLOC+20                       ; TEST INSTRUCTION
12077 057542  170203                    STFPS   R3                      ;GET FPS
12078 057544  012705  003202            MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
12079 057550  012702  057662            MOV     #TS32D5,R2              ;
12080 057554  004767  000014            JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12081 057560  020327  000004            CMP     R3,#4                   ;CHECK FPS
12082 057564  001401                    BEQ     3$                      ;OK GO ON
12083 057566  104003                    ERROR   +3              ;FPP ERROR
12084                                   ;NO GO TO ERROR
12085 057570                    3$:
12086                           ;
12087 057570  000167  000076            JMP     FIN32
12088                           ;
12089 057574  012701  000004    CHEC32: MOV     #4,R1                   ;INIT COUNTER
12090 057600  022522            1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12091 057602  001401                    BEQ     2$                      ;YES GO ON
12092 057604  104003                    ERROR   +3              ;FPP ERROR
12093                                   ;NO GO TO ERROR
12094 057606  077104            2$:     SOB     R1,1$                   ;ARE WE DONE
12095 057610  000207                    RTS     PC                      ;RETURN
12096                           ;
12097 057612  170000    TS32D0: .WORD   170000
12098 057614  003541            .WORD   3541
12099 057616  177777            .WORD   177777
12100 057620  172710            .WORD   172710
12101 057622  070000    TS32D1: .WORD   70000
12102 057624  003541            .WORD   3541
12103 057626  177777            .WORD   177777
12104 057630  172710            .WORD   172710
12105 057632  000177            .WORD   177
12106 057634  100000            .WORD   100000
12107 057636  177777            .WORD   177777
12108 057640  177007            .WORD   177007
12109 057642  070000    TS32D3: .WORD   70000
12110 057644  003541            .WORD   3541
12111 057646  177777            .WORD   177777
12112 057650  172710            .WORD   172710
12113 057652  170000    TS32D4: .WORD   170000
12114 057654  003541            .WORD   3541
12115 057656  177777            .WORD   177777
```

FLOATING POINT TESTS

```
12116 057660  172710                    .WORD   172710
12117 057662  000000          TS32D5:   .WORD   0
12118 057664  000000                    .WORD   0
12119 057666  177777                    .WORD   177777
12120 057670  177007                    .WORD   177007
12121 057672  000240          FIN32:    NOP
12122                         ;
12125                         ;
12126                         ;---------------------------------
12127                         ;TEST  NEGD
12128                         ;
12129                         ;
12130 057674                  TSFP33:
12131                         ;
12132 057674  005037  135370            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12133 057700  012705  000200            MOV     #200,R5                 ;SETUP TO LOAD FPS
12134 057704  170105                    LDFPS   R5                      ;SET FD=1
12135 057706  012701  000014            MOV     #12.,R1                 ;INIT COUNTER
12136 057712  012704  003162            MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
12137 057716  012703  060066            MOV     #TS33D0,R3              ;SETUP POINTER TO TEST VALUE
12138 057722  012324          100$:     MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
12139 057724  077102                    SOB     R1,100$                 ;ARE WE DONE
12140 057726  170767  123230            NEGD    TSTLOC                  ; TEST INSTRUCTION
12141 057732  170203                    STFPS   R3                      ;GET FPS
12142 057734  012705  003162            MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
12143 057740  012702  060116            MOV     #TS33D3,R2              ;
12144 057744  004767  000100            JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12145 057750  020327  000200            CMP     R3,#200                 ;CHECK FPS
12146 057754  001401                    BEQ     1$                      ;OK GO ON
12147 057756  104003                    ERROR   +3              ;FPP ERROR
12148                                                           ;NO GO TO ERROR
12149 057760  170767  123206    1$:     NEGD    TSTLOC+10                       ; TEST INSTRUCTION
12150 057764  170203                    STFPS   R3                      ;GET FPS
12151 057766  012705  003172            MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
12152 057772  012702  060126            MOV     #TS33D4,R2              ;
12153 057776  004767  000046            JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12154 060002  020327  000210            CMP     R3,#210                 ;CHECK FPS
12155 060006  001401                    BEQ     2$                      ;OK GO ON
12156 060010  104003                    ERROR   +3              ;FPP ERROR
12157                                                           ;NO GO TO ERROR
12158 060012  170767  123164    2$:     NEGD    TSTLOC+20                       ; TEST INSTRUCTION
12159 060016  170203                    STFPS   R3                      ;GET FPS
12160 060020  012705  003202            MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
12161 060024  012702  060136            MOV     #TS33D5,R2              ;
12162 060030  004767  000014            JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12163 060034  020327  000204            CMP     R3,#204                 ;CHECK FPS
12164 060040  001401                    BEQ     3$                      ;OK GO ON
12165 060042  104003                    ERROR   +3              ;FPP ERROR
12166                                                           ;NO GO TO ERROR
12167 060044                  3$:
12168                         ;
12169 060044  000167  000076            JMP     FIN33
12170                         ;
12171 060050  012701  000004  CHEC33:   MOV     #4,R1                   ;INIT COUNTER
12172 060054  022522          1$:       CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12173 060056  001401                    BEQ     2$                      ;YES GO ON
12174 060060  104003                    ERROR   +3              ;FPP ERROR
```

```
                                                                        ;NO GO TO ERROR
060062  077104                  2$:     SOB     R1,1$                           ;ARE WE DONE
060064  000207                          RTS     PC                              ;RETURN
                                ;
060066  170000                  TS33D0: .WORD   170000
060070  003541                          .WORD   3541
060072  177777                          .WORD   177777
060074  172710                          .WORD   172710
060076  070000                  TS33D1: .WORD   70000
060100  003541                          .WORD   3541
060102  177777                          .WORD   177777
060104  172710                          .WORD   172710
060106  000177                  TS33D2: .WORD   177
060110  100000                          .WORD   100000
060112  177777                          .WORD   177777
060114  177007                          .WORD   177007
060116  070000                  TS33D3: .WORD   70000
060120  003541                          .WORD   3541
060122  177777                          .WORD   177777
060124  172710                          .WORD   172710
060126  170000                  TS33D4: .WORD   170000
060130  003541                          .WORD   3541
060132  177777                          .WORD   177777
060134  172710                          .WORD   172710
060136  000000                  TS33D5: .WORD   0
060140  000000                          .WORD   0
060142  000000                          .WORD   0
060144  000000                          .WORD   0
060146  000240                  FIN33:  NOP
                                ;
                                ;
                                ;
                                ;---------------------------------
                                ;TEST LDD MODE 0, ILLEGAL AC7
                                ;
                                        ;
                                        ;
                                        ;
060150                          MFSRCM0:

                                ;
060150  012704  047600                  MOV     #47600,R4                       ;SETUP FPP STATUS
060154  170104                          LDFPS   R4                              ;LOAD FP STATUS
060156  012702  003122                  MOV     #RECFEC,R2                      ;POINT TO RECEIVED FEC MEMORY
060162  172407                  1$:     LDD     R7,AC0                          ;*TEST INSTRUCTION
                                                                                ;LOAD AC0 FROM ILLEGAL AC7
060164  170201                          STFPS   R1                              ;SAVE FPP STATUS
060166  022701  147600                  CMP     #147600,R1                      ;VERIFY FER BIT SET
060172  001401                          BEQ     2$                              ;BRANCH IF GOOD ERROR CONDITION
060174  104003                          ERROR   +3                      ;FPP ERROR
                                                                        ;THE FER BIT DIDNT SET
060176  170312                  2$:     STST    (R2)                            ;SAVE FEC AND FEA
060200  022722  000002                  CMP     #2,(R2)+                        ;VERIFY FEC CONTENTS
060204  001401                          BEQ     3$                              ;BRANCH IF GOOD
060206  104003                          ERROR   +3                      ;FPP ERROR
                                                                        ;FEC NE 2 (OPCODE ERROR)
060210  022722  060162          3$:     CMP     #1$,(R2)+                       ;VERFIY FEA CONTENTS
```

K1

FLOATING POINT TESTS

```
12234 060214  001401                        BEQ     4$                      ;BRANCH FI GOOD
12235 060216  104003                        ERROR   .+3                ;FPP ERROR
12236                                                                   ;FEA NOT CORRECT ERROR ADDRESS
12237 060220                          4$:
12238
12239
12242                                 ;
12243                                 ;
12244                                 ;---------------------------------
12245                                 ;TEST LDD MODE2
12246                                 ;
12247                                         ;
12248                                         ;
12249                                         ;
12250 060220                          MLDDM2:
12251
12252                                 ;
12253 060220  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12254 060224  012704  003234                  MOV     #TAB1,R4                ;POINT TO GOOD DATA
12255 060230  012702  047750                  MOV     #47750,R2               ;LOAD GOOD STATUS
12256 060234  170102                          LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12257 060236  172424                          LDD     (R4)+,AC0               ;*TEST INSTRUCTION - MODE 2
12258 060240  170203                          STFPS   R3                      ;SAVE TEST FPP STATUS
12259 060242  174021                          STD     AC0,(R1)+               ;SAVE TEST RESULT MODE 2
12260 060244  020203                          CMP     R2,R3                   ;VERIFY FPP STATUS
12261 060246  001401                          BEQ     1$                      ;BRANCH IF GOOD
12262 060250  104003                          ERROR   .+3                ;FPP ERROR
12263                                                                   ;BAD FPP STSTUS
12264 060252  022704  003244          1$:     CMP     #TAB1+10,R4                     ;VERFIY AUTO-INCR
12265 060256  001401                          BEQ     2$                      ;BRANCH IF GOOD
12266 060260  104003                          ERROR   .+3                ;FPP ERROR
12267                                                                   ;BAD AUTO-INCR
12268 060262  012704  003234          2$:     MOV     #TAB1,R4                ;POINT TO RECEIVED DATA
12269 060266  162701  000010                  SUB     #10,R1                  ;RETURN R1 TO PROPER VALUE
12270 060272  004767  055116                  JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12271 060276  005767  122616                  TST     COUNT                   ;SEE IF COUNTER=0
12272 060302  001401                          BEQ     3$                      ;BRANCH IF GOOD COMPARE
12273 060304  104003                          ERROR   .+3                ;FPP ERROR
12274                                                                   ;BAD DATA FROM FPP
12275 060306                          3$:
12276
12277                                 ;
12280                                 ;
12281                                 ;
12282                                 ;
12283                                 ;---------------------------------
12284                                 ;TEST LDD MODE 3
12285                                 ;
12286                                         ;
12287                                         ;
12288                                         ;
12289 060306                          MLDDM3:
12290
12291                                 ;
12292 060306  012737  003142  003164          MOV     #RECDST,@#TSTLOC+2              ;POINT TO RECEIVED DATA LOCATION
12293 060314  012701  003164                  MOV     #TSTLOC+2,R1                    ;SETUP STD IN MODE 3
12294 060320  012737  003244  003162          MOV     #TAB2,@#TSTLOC          ;POINT TO DATA TABLE
```

FLOATING POINT TESTS

```
12295 060326  012704  003162            MOV     #TSTLOC,F4              ;POINT TO GOOD DATA
12296 060332  012702  047750            MOV     #47750,R2           ;LOAD GOOD STATUS
12297 060336  170102                    LDFPS   R2                  ;LOAD FPP STATUS - DOUBLE,ID
12298 060340  172434                    LDD     @(R4)+,AC0              ;*TEST INSTRUCTION - MODE 2
12299 060342  170203                    STFPS   R3                  ;SAVE TEST FPP STATUS
12300 060344  174031                    STD     AC0,@(R1)+              ;SAVE TEST RESULT IN MODE 3
12301 060346  022703  047740            CMP     #47740,R3               ;VERIFY FPP STATUS
12302 060352  001401                    BEQ     1$                  ;BRANCH IF GOOD
12303 060354  104003                    ERROR   +3              ;FPP ERROR
12304                                                           ;BAD FPP STSTUS
12305 060356  022704  003164      1$:   CMP     #TSTLOC+2,R4            ;VERFIY AUTO-INCR
12306 060362  001401                    BEQ     2$                  ;BAD AUTO-DEC ON LDD
12307 060364  104003                    ERROR   +3              ;FPP ERROR
12308                                                           ;BAD AUTO-INC
12309 060366  022701  003166      2$:   CMP     #TSTLOC+4,R1            ; TEST STD AUTO-INC
12310 060372  001401                    BEQ     3$                  ;BRANCH IF GOOD
12311 060374  104003                    ERROR   +3              ;FPP ERROR
12312                                                           ;BAD AUTO-INCR
12313 060376  012704  003244      3$:   MOV     #TAB2,R4                ;POINT TO RECEIVED DATA
12314 060402  012701  003142            MOV     #RECDST,R1              ;POINT TO RECEIVED DATA
12315 060406  004767  055002            JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12316 060412  005767  122502            TST     COUNT                   ;SEE IF COUNTER=0
12317 060416  001401                    BEQ     4$                  ;BRANCH IF GOOD COMPARE
12318 060420  104003                    ERROR   +3              ;FPP ERROR
12319                                                           ;BAD DATA FROM FPP
12320 060422                      4$:
12321
12322                                   ;
12323                                   ;
12326                                   ;
12327                                   ;
12328                                   ;-----------------------------------
12329                                   ;TEST LDF, STD MODE 4
12330                                   ;
12331                                           ;
12332                                           ;
12333                                           ;
12334 060422                      MLDDM4:
12335
12336                                   ;
12337 060422  012701  003146                    MOV     #RECDST+4,R1            ;POINT TO RECEIVED DATA LOCATION
12338 060426  012704  003260                    MOV     #TAB3+4,R4              ;POINT TO GOOD DATA
12339 060432  012705  003314                    MOV     #TAB6,R5                ;CLEAR OUT AC0
12340 060436  170127  000200                    LDFPS   #200                    ;SET TO DOUBLE
12341 060442  172415                            LDD     (R5),AC0                    ;AC0=0
12342 060444  012702  047550                    MOV     #47550,R2               ;LOAD GOOD STATUS FLOATING
12343 060450  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12344 060452  172444                            LDF     -(R4),AC0                   ;*TEST INSTRUCTION - MODE 4
12345 060454  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12346 060456  012702  047750                    MOV     #47750,R2               ;SET TO DOUBLE MODE
12347 060462  170102                            LDFPS   R2                      ;SET FPP TO DOUBLE
12348 060464  174041                            STD     AC0,-(R1)                   ;SAVE TEST RESULT
12349 060466  022703  047540                    CMP     #47540,R3                   ;VERIFY FPP STATUS
12350 060472  001401                            BEQ     1$                  ;BRANCH IF GOOD
12351 060474  104003                            ERROR   +3              ;FPP ERROR
12352                                                                   ;BAD FPP STSTUS
12353 060476  022704  003254      1$:           CMP     #TAB3,R4                    ;VERFIY AUTO-DEC
```

FLOATING POINT TESTS

```
12354 060502  001401                    BEQ     2$                      ;BRANCH IF GOOD
12355 060504  104003                    ERROR   .3              ;FPP ERROR
12356                                                           ;BAD AUTO-INCR
12357 060506  012704  003254      2$:     MOV     #TAB3,R4                        ;POINT TO RECEIVED DATA
12358 060512  004767  054676              JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12359 060516  005767  122376              TST     COUNT                   ;SEE IF COUNTER=0
12360 060522  001401                    BEQ     3$                      ;BRANCH IF GOOD COMPARE
12361 060524  104003                    ERROR   .3              ;FPP ERROR
12362                                                           ;BAD DATA FROM FPP
12363 060526                      3$:
12364
12365                             ;
12366                             ;
12369                             ;
12370                             ;
12371                             ;---------------------------------
12372                             ;TEST LDD MODE 5
12373                             ;
12374                                     ;
12375                                     ;
12376                                     ;
12377 060526                      MLDDM5:
12378
12379                             ;
12380 060526  012701  003142              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12381 060532  012704  003164              MOV     #TSTLOC+2,R4                    ;POINT TO GOOD DATA
12382 060536  012737  003234  003162      MOV     #TAB1,@#TSTLOC                  ;SET UP MODE 5 POINTER TO DATA
12383 060544  012702  047750              MOV     #47750,R2               ;LOAD GOOD STATUS
12384 060550  170102                      LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12385 060552  172454                      LDD     @-(R4),AC0              ;*TEST INSTRUCTION - MODE 5
12386 060554  170203                      STFPS   R3                      ;SAVE TEST FPP STATUS
12387 060556  174011                      STD     AC0,(R1)                ;SAVE TEST RESULT
12388 060560  020203                      CMP     R2,R3                   ;VERIFY FPP STATUS
12389 060562  001401                      BEQ     1$                      ;BRANCH IF GOOD
12390 060564  104003                      ERROR   .3              ;FPP ERROR
12391                                                           ;BAD FPP STATUS
12392 060566  022704  003162      1$:     CMP     #TSTLOC,R4                      ;VERFIY AUTO-DEC
12393 060572  001401                      BEQ     2$                      ;BRANCH IF GOOD
12394 060574  104003                      ERROR   .3              ;FPP ERROR
12395                                                           ;BAD AUTO-DEC
12396 060576  012704  003234      2$:     MOV     #TAB1,R4                        ;POINT TO EXPECTED DATA
12397 060602  004767  054606              JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12398 060606  005767  122306              TST     COUNT                   ;SEE IF COUNTER=0
12399 060612  001401                      BEQ     3$                      ;BRANCH IF GOOD COMPARE
12400 060614  104003                      ERROR   .3              ;FPP ERROR
12401                                                           ;BAD DATA FROM FPP
12402 060616                      3$:
12403
12406                             ;
12407                             ;
12408                             ;---------------------------------
12409                             ;TEST LDD MODE 6
12410                             ;
12411                                     ;
12412                                     ;
12413                                     ;
12414 060616                      MLDDM6:
```

FLOATING POINT TESTS

```
12415
12416                                   ;
12417 060616  012701  003342                    MOV     #RECDST+200,R1          ;POINT TO RECEIVED DATA LOCATION
12418 060622  012704  003044                    MOV     #TAB2-200,R4                    ;SETUP R4 FOR MODE 6
12419 060626  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12420 060632  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12421 060634  172464  000200                    LDD     200(R4),AC0             ;LDD MODE 6
12422 060640  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12423 060642  174061  177600                    STD     AC0,-200(R1)                    ;SAVE TEST RESULT
12424 060646  022703  047740                    CMP     #47740,R3                       ;VERIFY FPP STATUS
12425 060652  001401                            BEQ     1$                      ;BRANCH IF GOOD
12426 060654  104003                            ERROR   +3                      ;FPP ERROR
12427                                                                           ;BAD FPP STATUS
12428 060656  162701  000200              1$:   SUB     #200,R1                         ;R1=RECDST
12429 060662  062704  000200              2$:   ADD     #200,R4                 ;POINT TO EXPECTED DATA
12430 060666  004767  054522                    JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12431 060672  005767  122222                    TST     COUNT                   ;SEE IF COUNTER=0
12432 060676  001401                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12433 060700  104003                            ERROR   +3                      ;FPP ERROR
12434                                                                           ;BAD DATA FROM FPP
12435 060702                              3$:
12436
12439                                   ;
12440                                   ;
12441                                   ;-----------------------------------
12442                                   ;TEST LDD MODE 7
12443                                   ;
12444                                           ;
12445                                           ;
12446                                           ;
12447 060702                            MLDDM7:
12448
12449                                   ;
12450 060702  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12451 060706  005004                            CLR     R4                      ;R4=0
12452 060710  012727  003234  003162              MOV     #TAB1,#TSTLOC           ;POINTER FOR MODE 7 GOOD DATA
12453 060716  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12454 060722  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12455 060724  172474  003162                    LDD     @TSTLOC(R4),AC0         ;*TEST INSTRUCTION - MODE 7
12456 060730  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12457 060732  174011                            STD     AC0,(R1)                        ;SAVE TEST RESULT
12458 060734  020203                            CMP     R2,R3                   ;VERIFY FPP STATUS
12459 060736  001401                            BEQ     1$                      ;BRANCH IF GOOD
12460 060740  104003                            ERROR   +3                      ;FPP ERROR
12461                                                                           ;BAD FPP STATUS
12462 060742  005704                      1$:   TST     R4                              ;VERFIY CONTENTS OF R4
12463 060744  001401                            BEQ     2$                      ;BRANCH IF GOOD
12464 060746  104003                            ERROR   +3                      ;FPP ERROR
12465                                                                           ;BAD R4
12466 060750  012704  003234              2$:   MOV     #TAB1,R4                        ;POINT TO RECEIVED DATA
12467 060754  004767  054434                    JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12468 060760  005767  122134                    TST     COUNT                   ;SEE IF COUNTER=0
12469 060764  001401                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12470 060766  104003                            ERROR   +3                      ;FPP ERROR
12471                                                                           ;BAD DATA FROM FPP
12472 060770                              3$:
12473
```

FLOATING POINT TESTS

```
12476                                   ;
12477                                   ;
12478                                   ;---------------------------------
12479                                   ;TEST LDD MODE 27 - ONLY 16 BITS ARE LOADED OR STORED
12480                                   ;
12481                                           ;
12482                                           ;
12483                                           ;
12484 060770                            MLDM27:
12485
12486                                   ;
12487 060770  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12488 060774  012704  003274                    MOV     #TAB5,R4                        ;POINT TO GOOD DATA
12489 061000  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12490 061004  005005                            CLR     R5                      ;R5=0
12491 061006  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12492 061010  172427  043243                    LDD     #5205,AC0               ;*TEST INSTRUCTION - MODE 27
12493 061014  005205                            INC     R5
12494 061016  005205                            INC     R5
12495 061020  005205                            INC     R5                      ; TEST PROPER PC PATH
12496 061022  022705  000003                    CMP     #3,R5                   ;VERIFY ONLY 3 PC INCREMENT
12497 061026  001401                            BEQ     1$                      ;BRANCH IF PROPER PC ACTION
12498 061030  104003                            ERROR   .3              ;FPP ERROR
12499                                                                   ;BAD MODE 27 LOAD
12500 061032  170203                    1$:     STFPS   R3                      ;SAVE TEST FPP STATUS
12501 061034  174011                            STD     AC0,(R1)                        ;SAVE TEST RESULT
12502 061036  022703  047740                    CMP     #47740,R3                       ;VERIFY FPP STATUS
12503 061042  001401                            BEQ     2$                      ;BRANCH IF GOOD
12504 061044  104003                            ERROR   .3              ;FPP ERROR
12505                                                                   ;BAD FPP STATUS
12506 061046  004767  054342            2$:     JSR     R7,DATVER                       ;VERFIY DATA FROM FPP
12507 061052  005767  122042                    TST     COUNT                           ;SEE IF COUNTER=0
12508 061056  001401                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12509 061060  104003                            ERROR   .3              ;FPP ERROR
12510                                                                   ;BAD DATA FROM FPP
12511 061062                            3$:
12512
12515                                   ;
12516                                   ;
12517                                   ;---------------------------------
12518                                   ;TEST ADDF, ADDD, SUBF, SUBD - AC0=0 FSRC=0;
12519                                           ;
12520                                           ;
12521                                           ;
12522 061062                            MNORM1:
12523
12524                                   ;
12525 061062  012704  003314                    MOV     #TAB6,R4                ;POINT TO FSRC TEST DATA
12526 061066  005067  122054                    CLR     RECDST+4                ;CLEAR OUT RECEIVED DATA TABLE
12527 061072  005067  122052                    CLR     RECDST+6                ;
12528 061076  012702  040000                    MOV     #40000,R2               ;SET UP GOOD STATUS
12529 061102  170102                            LDFPS   R2                      ;LOAD FPP STATUS, FLOATING
12530 061104  172414                            LDF     (R4),AC0                ;LOAD AC0 WITH 0
12531 061106  172014                            ADDF    (R4),AC0                ;0+0
12532 061110  170203                            STFPS   R3                      ;SAVE STATUS
12533 061112  022703  040004                    CMP     #40004,R3               ;VERIFY STATUS
12534 061116  001401                            BEQ     1$                      ;BRANCH IF GOOD
```

FLOATING POINT TESTS

```
12535 061120  104003                    ERROR   .3              ;FPP ERROR
12536                                                           ;BAD FPP STATUS
12537 061122  012701  003142      1$:     MOV     @RECDST,R1              ;POINT TO RECEIVERD DATA
12538 061126  174011                    STF     AC0,(R1)                ;SAVE DATA
12539 061130  004767  054260            JSR     R7,DATVER               ;VERIFY DATA
12540 061134  005767  121760            TST     COUNT                   ;
12541 061140  001401                    BEQ     2$                      ;BRANCH IF GOOD
12542 061142  104003                    ERROR   .3              ;FPP ERROR
12543                                                           ;BAD DATA IN AC0
12544 061144  012702  040200      2$:     MOV     @40200,R2               ;LOAD FLOATING STATUS
12545 061150  170102                    LDFPS   R2                      ;
12546 061152  172414                    LDD     (R4),AC0                ;LOAD AC0 WITH 0
12547 061154  172014                    ADDD    (R4),AC0                ;*TEST INSTRUCTION
12548 061156  174011                    STD     AC0,(R1)                ;SAVE DATA
12549 061160  170203                    STFPS   R3                      ;SAVE FPS
12550 061162  022703  040204            CMP     @40204,R3               ;VERFIY STATUS
12551 061166  001401                    BEQ     3$                      ;BRANCH IF GOOD
12552 061170  104003                    ERROR   .3              ;FPP ERROR
12553                                                           ;BAD FPS
12554 061172  004767  054216      3$:     JSR     R7,DATVER                       ;VERFIY DATA
12555 061176  005737  003120            TST     @@COUNT                 ;VERIFY RESULT
12556 061202  001401                    BEQ     44$                     ;BRANCH IF GOOD
12557 061204  104003                    ERROR   .3              ;FPP ERROR
12558                                                           ;BAD AC0
12559 061206  172414      44$:    LDD     (R4),AC0                ;SETUP DATA
12560 061210  173014                    SUBD    (R4),AC0                ;*TEST INSTRUCTION
12561 061212  170203                    STFPS   R3                      ;SAVE STATUS
12562 061214  022703  040204            CMP     @40204,R3                       ;VERFIY STATUS
12563 061220  001401                    BEQ     4$                      ;BRANCH IF GOOD
12564 061222  104003                    ERROR   .3              ;FPP ERROR
12565                                                           ;BAD FPS
12566 061224  174011      4$:     STD     AC0,(R1)                ;SAVE AC0 DATA
12567 061226  004767  054162            JSR     R7,DATVER                       ;VERFIY DATA
12568 061232  005767  121662            TST     COUNT                   ;
12569 061236  001401                    BEQ     5$                      ;BRANCH IF GOOD
12570 061240  104003                    ERROR   .3              ;FPP ERROR
12571                                                           ;BAD AC0
12572 061242  170127  000000      5$:     LDFPS   @0                      ;STORE FPP STATUS
12573 061246  172414                    LDD     (R4),AC0                ;LOAD AC0
12574 061250  173014                    SUBF    (R4),AC0                        ;0-0
12575 061252  170203                    STFPS   R3                      ;SAVE STATUS
12576 061254  174011                    STD     AC0,(R1)                ;SAVE AC0
12577 061256  022703  000004            CMP     @4,R3                   ;VERFIY STATUS
12578 061262  001401                    BEQ     6$                      ;BRANCH IF GOOD
12579 061264  104003                    ERROR   .3              ;FPP ERROR
12580                                                           ;BAD FPS
12581 061266  004767  054122      6$:     JSR     R7,DATVER                       ;VERIFY DATAT
12582 061272  005767  121622            TST COUNT                       ;
12583 061276  001401                    BEQ     7$              ;BRANC IF GOOD
12584 061300  104003                    ERROR   .3              ;FPP ERROR
12585                                                           ;BAD AC0
12586 061302      7$:
12587
12588
12591                             ;
12592                             ;
12593                             ;---------------------------------
```

D2

FLOATING POINT TESTS

```
12594                                   ;TEST ADDF,SUBD - FSRC=0, ACO NE 0
12595                                   ;
12596                                           ;
12597                                           ;
12598                                           ;
12599 061302                            MNNRM2:
12600
12601                                   ;
12602 061302  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12603 061306  012705  003314                    MOV     #TAB6,R5                ;POINT TO SOURCE DATA TABLE
12604 061312  012704  003244                    MOV     #TAB2,R4                ;POINT TO ACO DATA
12605 061316  170127  000200                    LDFPS   #200                    ;SET TO DOUBLE FOR CLEAR
12606 061322  172415                            LDD     (R5),AC0                ;
12607 061324  005002                            CLR     R2                      ;SETUP FPP STATUS
12608 061326  170102                            LDFPS   R2                      ;LOAD FPS
12609 061330  172414                            LDF     (R4),AC0                ;LOAD ACO
12610 061332  172015                            ADDF    (R5),AC0                        ;*TEST INSTRUCTION
12611 061334  170203                            STFPS   R3                      ;SAVE STATUS
12612 061336  174011                            STF     AC0,(R1)                ;SAVE ACO
12613 061340  022703  000000                    CMP     #0,R3                   ;VERFIY NEGATIVE RESULT
12614 061344  001401                            BEQ     1$                      ;BRANCH IF GOOD
12615 061346  104003                            ERROR   .3              ;FPP ERROR
12616                                                                   ;BAD FPS
12617 061350  012704  003264            1$:     MOV     #TAB4,R4                ;POINT TO EXPECTED DATA
12618 061354  004767  054034                    JSR     R7,DATVER                       ;VERFIY ACO
12619 061360  005767  121534                    TST     COUNT                   ;CHECK RESULT
12620 061364  001401                            BEQ     2$                      ;BRANCH IF GOOD
12621 061366  104003                            ERROR   .3              ;FPP ERROR
12622                                                                   ;BAD ACO
12623 061370  170127  000200            2$:     LDFPS   #200            ;SET STATUS TO DUOBLE NODE
12624 061374  172414                            LDD     (R4),AC0                ;LOAD ACO WITH A VALUE
12625 061376  173015                            SUBD    (R5),AC0                ;*TEST INSTRUCTION
12626 061400  170203                            STFPS   R3                      ;SAVE FPP STATUS
12627 061402  174011                            STD     AC0,(R1)                ;SAVE ACO
12628 061404  022703  000200                    CMP     #200,R3         ;VERIFY RESULT
12629 061410  001401                            BEQ     3$                      ;BRANCH IF GOOD
12630 061412  104003                            ERROR   .3              ;FPP ERROR
12631                                                                   ;BAD SUBD
12632 061414  012704  003264            3$:     MOV     #TAB4,R4                ;POINT TO EXPECTED
12633 061420  004767  053770                    JSR     R7,DATVER               ;VERIFY ACO
12634 061424  005767  121470                    TST     COUNT                   ;
12635 061430  001401                            BEQ     4$                      ;BRANCH IF GOOD ACO
12636 061432  104003                            ERROR   .3              ;FPP ERROR
12637                                                                   ;BAD ACO
12638 061434                            4$:
12639
12640
12643                                   ;
12644                                   ;
12645                                   ;-----------------------------------
12646                                   ;TEST ADDD, SUBF - FSRC NE 0, AC0=0
12647                                   ;
12648                                           ;
12649                                           ;
12650                                           ;
12651 061434                            MNNRM3:
12652
```

FLOATING POINT TESTS

```
12653                                   ;
12654 061434  012701  003142              MOV     #RECDST,R1          ;POINT TO RECEIVED DATA TABLE
12655 061440  012705  003314              MOV     #TAB6,R5            ;POINT TO ACO DATA TABLE
12656 061444  012704  003234              MOV     #TAB1,R4            ;POINT TO FSRC DATA
12657 061450  012702  000200              MOV     #200,R2             ;SETUP FPP STATUS
12658 061454  170102                      LDFPS   R2                  ;LOAD FPS
12659 061456  172415                      LDD     (R5),AC0            ;LOAD ACO
12660 061460  172014                      ADDD    (R4),AC0                    ;*TEST INSTRUCTION
12661 061462  170203                      STFPS   R3                  ;SAVE STATUS
12662 061464  174011                      STD     AC0,(R1)            ;SAVE ACO
12663 061466  022703  000210              CMP     #210,R3             ;VERFIY NEGATIVE RESULT
12664 061472  001401                      BEQ     1$                  ;BRANCH IF GOOD
12665 061474  104003                      ERROR   .3              ;FPP ERROR
12666                                                             ;BAD FPS
12667 061476  004767  053712        1$:   JSR     R7,DATVER                   ;VERFIY ACO
12668 061502  005767  121412              TST     COUNT               ;CHECK RESULT
12669 061506  001401                      BEQ     2$                  ;BRANCH IF GOOD
12670 061510  104003                      ERROR   .3              ;FPP ERROR
12671                                                             ;BAD ACO
12672 061512  170127  000200        2$:   LDFPS   #200                ;SET STATUS TO DUOBLE NODE
12673 061516  172415                      LDF     (R5),AC0            ;LOAD ACO WITH A VALUE
12674 061520  173014                      SUBD    (R4),AC0            ;*TEST INSTRUCTION
12675 061522  170203                      STFPS   R3                  ;SAVE FPP STATUS
12676 061524  174011                      STF     AC0,(R1)                    ;SAVE ACO
12677 061526  022703  000200              CMP     #200,R3             ;VERIFY RESULT
12678 061532  001401                      BEQ     3$                  ;BRANCH IF GOOD
12679 061534  104003                      ERROR   .3              ;FPP ERROR
12680                                                             ;BAD SUBD
12681 061536  012704  003504        3$:   MOV     #TAB18,R4                   ;POINT TO EXPECTED DATA
12682 061542  004767  053646              JSR     R7,DATVER           ;VERIFY ACO
12683 061546  005767  121346              TST     COUNT               ;
12684 061552  001401                      BEQ     4$                  ;BRANCH IF GOOD ACO
12685 061554  104003                      ERROR   .3              ;FPP ERROR
12686                                                             ;BAD ACO
12687 061556                        4$:
12688
12689
12692                                   ;
12693                                   ;
12694                                   ;-----------------------------------
12695                                   ;TEST  ADDF, SUBD - EXP(ACO) = EXP(FSRC)
12696                                   ;
12697                                           ;
12698                                           ;
12699                                           ;
12700 061556                        MNNRM4:
12701
12702                                   ;
12703 061556  012702  003240              MOV     #3240,R2            ;SET FIU,FD,FT
12704 061562  170102                      LDFPS   R2
12705 061564  012704  003334              MOV     #TAB7,R4                    ;SET FSRC
12706 061570  012705  003344              MOV     #TAB8,R5            ;SETUP ACO
12707 061574  012701  003142              MOV     #RECDST,R1          ;POINT TO RECEIVED DATA
12708 061600  172415                      LDD     (R5),AC0                    ;LOAD ACO
12709 061602  172014                      ADDD    (R4),AC0            ;*TEST INSTRUCTION
12710 061604  174011                      STD     AC0,(R1)            ;SAVE TEST RESULT
12711 061606  012704  003354              MOV     #TAB9,R4            ;POINT TO EXPECTED DATA
```

FLOATING POINT TESTS

```
12712 061612  004767  053576                JSR     R7,DATVER                       ;VERIFY ACO DATA
12713 061616  005767  121276                TST     COUNT                           ;
12714 061622  001401                        BEQ     1$                              ;BRANCH IF GOOD
12715 061624  104003                        ERROR   .3                      ;FPP ERROR
12716                                                                       ;BAD ADDD
12717 061626  012704  003344          1$:   MOV     #TAB8,R4                        ;
12718 061632  012703  003344                MOV     #TAB8,R3                        ;SETUP SAME ACO
12719 061636  012702  003200                MOV     #3200,R2                        ;ROUND MODE
12720 061642  170102                        LDFPS   R2
12721 061644  172413                        LDD     (R3),AC0                        ;LOAD AC0
12722 061646  061400                        ADD     (R4),AC0                        ;*TEST INSTRUCTION
12723 061650  174011                        STD     AC0,(R1)                        ;SAVE DATA
12724 061652  004767  053536                JSR     R7,DATVER                       ;VERIFY AC0
12725 061656  005767  121236                TST     COUNT
12726 061662  001401                        BEQ     2$                              ;BRANCH IF GOOD
12727 061664  104003                        ERROR   .3                      ;FPP ERROR
12728                                                                       ;BAD ROUND RESULT
12729 061666                          2$:
12730
12731
12734                                       ;
12735                                       ;
12736                                       ;-----------------------------------
12737                                       ;TEST ADD - EXP(FSRC) > EXP(AC0)
12738                                       ;
12739                                               ;
12740                                               ;
12741                                               ;
12742 061666                          MXDF1:
12743
12744                                       ;
12745 061666  012702  003200                        MOV     #3200,R2                ;R2=FPP STATUS
12746 061672  170102                        LDFPS   R2                              ;LOAD FPS STATUS
12747 061674  012704  003374                MOV     #TAB11,R4                       ;POINT TO FSRC DATA
12748 061700  012701  003142                MOV     #RECDST,R1                      ;POINT TO AC0 RESULT
12749 061704  012705  003364                MOV     #TAB10,R5                       ;POINT TO AC0 DATA
12750 061710  172415                        LDD     (R5),AC0                        ;LOAD AC0 DATA
12751 061712  172014                        ADDD    (R4),AC0                        ;*TEST INSTRUCTIONS
12752 061714  170203                        STFPS   R3                              ;SAVE FPP STATUS
12753 061716  174011                        STD     AC0,(R1)                        ;SAVE AC0 DATA
12754 061720  022703  003200                CMP     #3200,R3                        ;VERIFY FPP STATUS
12755 061724  001401                        BEQ     1$                              ;BRANCH IF GOOD
12756 061726  104003                        ERROR   .3                      ;FPP ERROR
12757                                                                       ;BAD FPP STATUS
12758 061730  012704  003404          1$:   MOV     #TAB11A,R4                      ;POINT TO EXPECTED DATA
12759 061734  004767  053454                JSR     R7,DATVER                       ;VERIFY CONTENTS OF AC0
12760 061740  005767  121154                TST     COUNT                           ;
12761 061744  001401                        BEQ     2$                              ;BRANCH IF GOOD AC0
12762 061746  104003                        ERROR   .3                      ;FPP ERROR
12763                                                                       ;BAD AC0, SHOULD = FSRC
12764 061750  012704  003414          2$:   MOV     #TAB12,R4                       ;POINT TO FSRC DATA
12765 061754  172415                        LDD     (R5),AC0                        ;AC0
12766 061756  172014                        ADDD    (R4),AC0                        ;*TEST INSTRUCTION
12767 061760  012704  003434                MOV     #TAB13B,R4                      ;POINT TO EXPECTED RESULT
12768 061764  174011                        STD     AC0,(R1)                        ;SAVE AC0 DATA INTO RECDAT
12769 061766  004767  053422                JSR     R7,DATVER                       ;VERIFY DATA
12770 061772  005767  121122                TST     COUNT
```

FLOATING POINT TESTS

```
12771 061776  001401                    BEQ     3$                      ;BRANCH IF GOOD DATA
12772 062000  104003                    ERROR   .3                  ;FPP ERROR
12773                                                               ;BAD ACO DATA
12774 062002  012702  003000      3$:     MOV     #3000,R2                ;GET FPP STATUS DATA
12775 062006  012704  003444            MOV     #TAB14,R4               ;POINT TO FSRC DATA
12776 062012  012705  003454            MOV     #TAB15,R5               ;POINT TO ACO DATA
12777 062016  172415                    LDD     (R5),AC0                ;LOAD AC0
12778 062020  170102                    LDFPS   R2                      ;FPP STATUS = FLOAT, INTERRUPTS ENABLE
12779 062022  172014                    ADDF    (R4),AC0                ;*TEST INSTRUCTION
12780 062024  170127  000200            LDFPS   #200                    ;RESET TO DOUBLE
12781 062030  174011                    STD     AC0, (R1)               ;RECDST=AC0
12782 062032  012704  003364            MOV     #TAB10,R4               ;POINT TO GOOD DATA
12783 062036  004767  053352            JSR     R7,DATVER                       ;VERFIY CONTENTS OF AC0
12784 062042  005767  121052            TST     COUNT                   ;
12785 062046  001401                    BEQ     4$                      ;BRANCH IF GOOD
12786 062050  104003                    ERROR   .3                  ;FPP ERROR
12787                                                               ;BAD FLOATING ADD
12788 062052  012705  003464      4$:     MOV     #TAB16,R5               ;POINT TO ACO DATA
12789 062056  170102                    LDFPS   R2                      ;FPP STATUS = FLOAT
12790 062060  172415                    LDF     (R5),AC0                ;LOAD AC0
12791 062062  172014                    ADDF    (R4),AC0                ;*TEST INSTRUCTION
12792 062064  174011                    STD     AC0,(R1)                        ;SAVE AC0  DATA
12793 062066  012704  003474            MOV     #TAB17,R4               ;POINT TO GOOD DATA
12794 062072  004767  053316            JSR     R7,DATVER
12795 062076  005767  121016            TST     COUNT
12796 062102  001401                    BEQ     5$                      ;BRANCH IF GOOD
12797 062104  104003                    ERROR   .3                  ;FPP ERROR
12798                                                               ;BAD FLOATING ADD
12799 062106                      5$:
12800
12801
12802
12805                             ;
12806                             ;
12807                             ;-----------------------------------
12808                             ;TEST ADDD WITH NEGATIVE OPERANDS
12809                             ;
12810                                     ;
12811                                     ;
12812                                     ;
12813 062106                      MNGOP:
12814
12815                             ;
12816 062106  012702  003200              MOV     #3200,R2                ;LOAD FPS VALUE
12817 062112  170102                      LDFPS   R2                      ;
12818 062114  012704  003514              MOV     #TAB21,R4               ;DATA ADDRESS FOR AC0 AND FSR
12819 062120  172414                      LDD     (R4),AC0                ;AC0=100200 0 0 0
12820 062122  172014                      ADDD    (R4),AC0                ;*TEST INSTRUCTION
12821 062124  170203                      STFPS   R3                      ;SAVE STATUS
12822 062126  012701  003142              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12823 062132  174011                      STD     AC0,(R1)                ;SAVE AC0 DATA
12824 062134  022703  003210              CMP     #3210,R3                ;VERIFY STATUS
12825 062140  001401                      BEQ     1$                      ;BRANCH IF GOOD
12826 062142  104003                      ERROR   .3                  ;FPP ERROR
12827                                                               ;
12828 062144  012704  003524      1$:     MOV     #TAB22,R4               ;POINT TO EXPECTED DATA
12829 062150  004767  053240              JSR     R7,DATVER               ;
```

FLOATING POINT TESTS

```
12830 062154  005767  120740            TST     COUNT                   ;VERIFY DATA
12831 062160  001401                    BEQ     2$                      ;BRANCH IF GOOD
12832 062162  104003                    ERROR   .3              ;FPP ERROR
12833                                                           ;
12834                                   ;       !-FSRC! = !ACO!
12835 062164  012704  003514      2$:   MOV     #TAB21,R4               ;POINT TO FSRC DATA
12836 062170  012701  003534            MOV     #TAB23,R1               ;POINT TO ACO DATA
12837 062174  012737  062212  000244    MOV     #101$,@#FPVEC           ;SETUP FP VECTOR
12838 062202  172411                    LDD     (R1),AC0                ;LOAD AC0
12839 062204  172014                    ADDD    (R4),AC0                ;*TEST INSTRUCTION
12840 062206  170000      100$:         CFCC                            ;COPY FPP CC
12841 062210  104003                    ERROR   .3              ;FPP ERROR
12842                                                           ;GO TO ERROR
12843 062212  170203      101$:         STFPS   R3                      ;SAVE FPP STATUS
12844 062214  012701  003142            MOV     #RECDST,R1                      ;POINT TO RECEIVED DATA TABLE
12845 062220  174011                    STD     AC0,(R1)                ;SAVE AC0 DATA
12846 062222  022703  103200            CMP     #103200,R3              ;VERIFY STATUS
12847 062226  001401                    BEQ     3$                      ;BRANCH IF GOOD
12848 062230  104003                    ERROR   .3              ;FPP ERROR
12849                                                           ;BAD STATUS
12850 062232  012605      3$:           MOV     (SP)+,R5                ;GET ERROR PC
12851 062234  020527  062206            CMP     R5,#100$                ;VERIFY ERROR ADDRESS ON STACK
12852 062240  001401                    BEQ     102$                    ;BRANCH IF GOOD
12853 062242  104003                    ERROR   .3              ;FPP ERROR
12854                                                           ;BAD ERROR RETURN ON STACK
12855 062244  005726      102$:         TST     (SP)+                   ;RESTORE STACK
12856 062246  012704  003544            MOV     #TAB24,R4               ;POINT TO EXPECTED DATA TABLE
12857 062252  004767  053136            JSR     R7,DATVER               ;VERIFY DATA
12858 062256  005767  120636            TST     COUNT
12859 062262  001401                    BEQ     4$                      ;BRANC IF GOOD
12860 062264  104003                    ERROR   .3              ;FPP ERROR
12861                                                           ;BAD AC0 DATA
12862                                   ;       !-AC0! = !FSRC!
12863 062266  012704  003534      4$:   MOV     #TAB23,R4               ;POINT TO FSRC DATA
12864 062272  012701  003514            MOV     #TAB21,R1               ;POINT TO ACO DATA
12865 062276  012737  062322  000244    MOV     #104$,@#FPVEC           ;SETUP FP VECTOR
12866 062304  012702  003200            MOV     #3200,R2                ;LOAD FPS VALUE
12867 062310  170102                    LDFPS   R2                      ;
12868 062312  172411                    LDD     (R1),AC0                ;LOAD AC0 DATA
12869 062314  172014                    ADDD    (R4),AC0                ;*TEST INSTRUCTION
12870 062316  170000      103$:         CFCC                            ;COPY FPP CC
12871 062320  104003                    ERROR   .3              ;FPP ERROR
12872                                                           ;GO TO ERROR
12873 062322  170203      104$:         STFPS   R3                      ;SAVE FPS
12874 062324  012701  003142            MOV     #RECDST,R1              ;SAVE ACO
12875 062330  174011                    STD     AC0,(R1)                ;
12876 062332  022703  103200            CMP     #103200,R3              ;VERFIY STATUS
12877 062336  001401                    BEQ     5$                      ;BRANCH IF GOOD
12878 062340  104003                    ERROR   .3              ;FPP ERROR
12879                                                           ;BAD FPS STATUS
12880 062342  012605      5$:           MOV     (SP)+,R5                ;GET ERROR PC
12881 062344  020527  062316            CMP     R5,#103$                ;VERIFY ERROR ADDRESS ON STACK
12882 062350  001401                    BEQ     105$                    ;BRANCH IF GOOD
12883 062352  104003                    ERROR   .3              ;FPP ERROR
12884                                                           ;BAD ERROR RETURN ON STACK
12885 062354  005726      105$:         TST     (SP)+                   ;RESTORE STACK
12886 062356  012704  003544            MOV     #TAB24,R4               ;POINT TO EXPECTED DATA
```

FLOATING POINT TESTS

```
12887 062362  004767  053026                JSR     R7,DATVER
12888 062366  005767  120526                TST     COUNT                   ;
12889 062372  001401                        BEQ     6$                      ;BRANCH IF  GOOD
12890 062374  104003                        ERROR   .+3                 ;FPP ERROR
12891                                                                   ;BAD ACO
12892                                       ;       !-FSRC! <           !ACO!
12893 062376  012704  003564          6$:   MOV     #TAB26,R4               ;POINT TO FSRC DATA
12894 062402  012701  003554                MOV     #TAB25,R1               ;POINT TO ACO DATA
12895 062406  012702  003200                MOV     #3200,R2                ;LOAD FPS VALUE
12896 062412  170102                        LDFPS   R2                      ;
12897 062414  012737  000246  000244        MOV     #246,@#FPVEC            ;SETUP FP VECTOR
12898 062422  172411                        LDD     (R1),AC0                ;LOAD ACO DATA
12899 062424  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
12900 062426  170203                        STFPS   R3                      ;SAVE STATUS
12901 062430  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12902 062434  174011                        STD     AC0,(R1)                        ;SAVE ACO
12903 062436  020327  003200                CMP     R3,#3200                        ;VERIFY STATUS
12904 062442  001401                        BEQ     7$                      ;BRANCH IF GOOD
12905 062444  104003                        ERROR   .+3                 ;FPP ERROR
12906                                                                   ;BAD FPS
12907 062446  012704  003574          7$:   MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
12908 062452  004767  052736                JSR     R7,DATVER                       ;VERIFY DATA
12909 062456  005767  120436                TST     COUNT                   ;
12910 062462  001401                        BEQ     8$                      ;BRANCH IF GOOD
12911 062464  104003                        ERROR   .+3                 ;FPP ERROR
12912                                                                   ;
12913                                       ;       !FSRC!  >           !-AC!
12914 062466  012704  003554          8$:   MOV     #TAB25,R4               ;POINT TO FSRC DATA
12915 062472  012701  003564                MOV     #TAB26,R1               ;POINT TO ACO DATA
12916 062476  172411                        LDD     (R1),AC0                ;LOAD ACO DATA
12917 062500  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
12918 062502  170203                        STFPS   R3                      ;SAVE STATUS
12919 062504  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12920 062510  174011                        STD     AC0,(R1)                        ;SAVE ACO
12921 062512  020327  003200                CMP     R3,#3200                        ;VERIFY STATUS
12922 062516  001401                        BEQ     9$                      ;BRANCH IF GOOD
12923 062520  104003                        ERROR   .+3                 ;FPP ERROR
12924                                                                   ;BAD FPS
12925 062522  012704  003574          9$:   MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
12926 062526  004767  052662                JSR     R7,DATVER                       ;VERIFY DATA
12927 062532  005767  120362                TST     COUNT                   ;
12928 062536  001401                        BEQ     10$                     ;BRANCH IF GOOD
12929 062540  104003                        ERROR   .+3                 ;FPP ERROR
12930                                                                   ;BAD ACO
12931                                       ;       !-FSRC! <           !ACO!
12932 062542  012704  003614          10$:  MOV     #TAB29,R4               ;POINT TO FSRC DATA
12933 062546  012701  003604                MOV     #TAB28,R1               ;POINT TO ACO DATA
12934 062552  172411                        LDD     (R1),AC0                ;LOAD ACO DATA
12935 062554  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
12936 062556  170203                        STFPS   R3                      ;SAVE STATUS
12937 062560  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12938 062564  174011                        STD     AC0,(R1)                        ;SAVE ACO
12939 062566  020327  003200                CMP     R3,#3200                        ;VERIFY STATUS
12940 062572  001401                        BEQ     11$                     ;BRANCH IF GOOD
12941 062574  104003                        ERROR   .+3                 ;FPP ERROR
12942                                                                   ;BAD FPS
12943 062576  012704  003624          11$:  MOV     #TAB29A,R4              ;POINT TO EXPECTED DATA
```

J2

FLOATING POINT TESTS

```
12944 062602  004767  052606                  JSR     R7,DATVER                       ;VERIFY DATA
12945 062606  005767  120306                  TST     COUNT                   ;
12946 062612  001401                          BEQ     12$                     ;BRANCH IF GOOD
12947 062614  104003                          ERROR   +3              ;FPP ERROR
12948                                                                 ;
12949 062616                          12$:
12950
12951                                 ;
12952                                 ;
12955                                 ;
12956                                 ;
12957                                 ;-----------------------------------
12958                                 ;TEST SUB WITH EXP[AC0]=EXP[FSRC]
12959                                 ;
12960                                         ;
12961                                         ;
12962                                         ;
12963 062616                          MSB:
12964
12965                                 ;
12966 062616  012702  003200                  MOV     #3200,R2                ;LOAD FPS DATA
12967 062622  170102                          LDFPS   R2                      ;LOAD FPS
12968 062624  012704  003514                  MOV     #TAB21,R4               ;POINT TO FSRC DATA
12969 062630  012701  003142                  MOV     #RECDST,R1              ;POINT TO AC0 RECEIVED DATA TABLE
12970 062634  172414                          LDD     (R4),AC0                ;LOAD AC0
12971 062636  173014                          SUBD    (R4),AC0                        ;*TEST INSTRUCTION
12972 062640  170203                          STFPS   R3                      ;SAVE STATUS
12973 062642  174011                          STD     AC0,(R1)                ;SAVE AC0 INTO RECDST
12974 062644  022703  003204                  CMP     #3204,R3                ;VERIFY STATUS
12975 062650  001401                          BEQ     1$                      ;BRANCH IF GOOD
12976 062652  104003                          ERROR   +3              ;FPP ERROR
12977                                                                 ;BAD FPS STATUS
12978 062654  012704  003314          1$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
12979 062660  004767  052530                  JSR     R7,DATVER               ;VERIFY AC0
12980 062664  005767  120230                  TST     COUNT                   ;
12981 062670  001401                          BEQ     2$                      ;BRANCH IF GOOD
12982 062672  104003                          ERROR   +3              ;FPP ERROR
12983                                                                 ;BAD AC0
12984 062674  012704  003444          2$:     MOV     #TAB14,R4               ;POINT TO FSRC AND AC0 DATA
12985 062700  172414                          LDD     (R4),AC0                        ;LOAD AC0 DATA
12986 062702  173014                          SUBD    (R4),AC0                        ;*TEST  INSTRUCTION
12987 062704  170203                          STFPS   R3                      ;SAVE FPS
12988 062706  174011                          STD     AC0,(R1)                                ;SAVE AC0 INTO RECDST
12989 062710  022703  003204                  CMP     #3204,R3                        ;VERIFY FPS
12990 062714  001401                          BEQ     3$                      ;BRANCH IF GOOD
12991 062716  104003                          ERROR   +3              ;FPP ERROR
12992                                                                 ;BAD AC0
12993 062720  012704  003314          3$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
12994 062724  004767  052464                  JSR     R7,DATVER                       ;VERIFY AC0
12995 062730  005767  120164                  TST     COUNT
12996 062734  001401                          BEQ     4$                      ;BRANCH IF GOOD
12997 062736  104003                          ERROR   +3              ;FPP ERROR
12998                                                                 ;BAD AC0
12999 062740                          4$:
13000
13001                                 ;
13004                                 ;
```

FLOATING POINT TESTS

```
13005                                   ;
13006                                   ;---------------------------------
13007                                   ;TEST NORMALIZE
13008                                   ;
13009                                           ;
13010                                           ;
13011                                           ;
13012 062740                            MNRM:
13013
13014                                   ;
13015 062740  012702  003200                    MOV     #3200,R2                ;LOAD FPS
13016 062744  170102                            LDFPS   R2                      ;
13017 062746  012705  003644                    MOV     #TAB31,R5               ;POINT TO FSRC DATA
13018 062752  012701  003634                    MOV     #TAB30,R1               ;POINT TO ACO DATA
13019 062756  172411                            LDD     (R1),AC0                ;LOAD AC0
13020 062760  173015                            SUBD    (R5),AC0                ;*TEST INSTRUCTION
13021                                                                           ;1 LEFT SHIFT
13022 062762  170203                            STFPS   R3                      ;SAVE STATUS
13023 062764  012704  003142                    MOV     #RECDST,R4              ;POINT TO RECDATA
13024 062770  174014                            STD     AC0,(R4)                        ;SAVE AC0
13025 062772  012701  003674                    MOV     #TAB34,R1               ;POINT TO EXPECTED DATA
13026 062776  004767  052412                    JSR     R7,DATVER                       ;VERIFY DATA
13027 063002  005767  120112                    TST     COUNT
13028 063006  001401                            BEQ     1$                      ;BRANCH IF GOOD
13029 063010  104003                            ERROR   +3              ;FPP ERROR
13030                                                                   ;
13031 063012  012701  003654            1$:     MOV     #TAB32,R1               ;AC0 DATA
13032 063016  012705  003664                    MOV     #TAB33,R5                       ;FSRC DATA
13033 063022  172411                            LDD     (R1),AC0                        ;LOAD AC0
13034 063024  173015                            SUBD    (R5),AC0                ;*TST INSTRUCTION
13035                                                                           ;56 LEFT SHIFTS
13036 063026  012701  003142                    MOV     #RECDST,R1                      ;SAVE DATA
13037 063032  174011                            STD     AC0,(R1)                        ;
13038 063034  004767  052354                    JSR     R7,DATVER
13039 063040  005767  120054                    TST     COUNT
13040 063044  001401                            BEQ     2$                      ;
13041 063046  104003                            ERROR   +3              ;FPP ERROR
13042                                                                   ;
13043 063050                            2$:
13044
13047                                   ;
13048                                   ;
13049                                   ;---------------------------------
13050                                   ;TEST ADDD WITH OVERFLOW AND UNDERFLOW
13051                                   ;
13052                                           ;
13053                                           ;
13054                                           ;
13055 063050                            MUVAD:
13056
13057                                   ;
13058 063050  012702  000200                    MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13059 063054  170102                            LDFPS   R2                      ;LOAD FPS
13060 063056  012704  003704                    MOV     #TAB40,R4                       ;POINT TO FSRC DATA
13061 063062  012701  003704                    MOV     #TAB40,R1                       ;POINT TO AC0 DATA
13062 063066  172411                            LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13063 063070  172014                            ADDD    (R4),AC0                ;*TEST INSTRUCTION
```

L2

FLOATING POINT TESTS

```
13064 063072  170203                    STFPS   R3                      ;SAVE FPS
13065 063074  012701  003142              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13066 063100  174011                    STD     AC0,(R1)                ;SAVE AC0  RESULT
13067 063102  022703  000206              CMP     #206,R3                 ;VERIFY STATUS
13068 063106  001401                    BEQ     1$                      ;BRANCH IF GOOD
13069 063110  104003                    ERROR   .+3             ;FPP ERROR
13070                                           ;BAD FPS
13071 063112  012704  003314      1$:   MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13072 063116  004767  052272            JSR     R7,DATVER                       ;VERIFY DATA
13073 063122  005767  117772            TST     COUNT
13074 063126  001401                    BEQ     2$                      ;BRANCH IF GOOD
13075 063130  104003                    ERROR   .+3             ;FPP ERROR
13076                                           ;BAD AC0
13077                           ;OVERFLOW TRAPS ENABLED
13078 063132  012702  001200      2$:   MOV     #1200,R2                ;SETUP FLOATING POINT STATUS
13079 063136  170102                    LDFPS   R2                      ;LOAD FPS
13080 063140  012704  003704            MOV     #TAB40,R4               ;POINT TO FSRC DATA
13081 063144  012701  003704            MOV     #TAB40,R1               ;POINT TO AC0 DATA
13082 063150  172411                    LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13083 063152  012737  063166  000244      MOV     #3$,@#FPVEC             ;CHANGE TRAP VECTOR
13084 063160  172014                    ADDD    (R4),AC0                ;*TEST INSTRUCTION
13085 063162  170000              23$:  CFCC
13086 063164  104003                    ERROR   .+3             ;FPP ERROR
13087                                           ;FAILED TO TRAP ON OVERFLOW
13088 063166  170203              3$:   STFPS   R3                      ;SAVE FPS
13089 063170  012701  003142            MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13090 063174  174011                    STD     AC0,(R1)                ;SAVE AC0  RESULT
13091 063176  022703  101206              CMP     #101206,R3                      ;VERIFY STATUS
13092 063202  001401                    BEQ     4$                      ;BRANCH IF GOOD
13093 063204  104003                    ERROR   .+3             ;FPP ERROR
13094                                           ;BAD FPS
13095 063206  012600              4$:   MOV     (SP)+,R0                ;CHECK STORED PC
13096 063210  022700  063162            CMP     #23$,R0                 ;
13097 063214  001401                    BEQ     5$                      ;BRANCH IF RETURN ADDRESS IS GOOD
13098 063216  104003                    ERROR   .+3             ;FPP ERROR
13099                                           ;BAD RETURN ADDRESS
13100 063220  012600              5$:   MOV     (SP)+,R0                ;CLEAN UP STACK
13101 063222  012704  003314            MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13102 063226  004767  052162            JSR     R7,DATVER                       ;VERIFY DATA
13103 063232  005767  117662            TST     COUNT
13104 063236  001401                    BEQ     7$                      ;BRANCH IF GOOD
13105 063240  104003                    ERROR   .+3             ;FPP ERROR
13106                                           ;BAD AC0
13107                           ;UNDERFLOW TRAPS DISABLED
13108 063242  012702  000200      7$:   MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13109 063246  170102                    LDFPS   R2                      ;LOAD FPS
13110 063250  012737  135360  000244      MOV     #WLDTRP,@#FPVEC                 ;REPLACE WILD TRAP VECTOR
13111 063256  012704  003334            MOV     #TAB7,R4                        ;POINT TO FSRC DATA
13112 063262  012701  003714            MOV     #TAB41,R1                       ;POINT TO AC0 DATA
13113 063266  172411                    LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13114 063270  172014                    ADDD    (R4),AC0                ;*TEST INSTRUCTION
13115 063272  170203                    STFPS   R3                      ;SAVE FPS
13116 063274  012701  003142            MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13117 063300  174011                    STD     AC0,(R1)                ;SAVE AC0  RESULT
13118 063302  022703  000204            CMP     #204,R3                 ;VERIFY STATUS
13119 063306  001401                    BEQ     8$                      ;BRANCH IF GOOD
13120 063310  104003                    ERROR   .+3             ;FPP ERROR
```

FLOATING POINT TESTS

```
13121
13122 063312  012704  003314          8$:     MOV     #TAB6,R4        ;BAD FPS
                                                                        ;POINT TO EXPECTED DATA
13123 063316  004767  052072                  JSR     R7,DATVER       ;VERIFY DATA
13124 063322  005767  117572                  TST     COUNT
13125 063326  001401                          BEQ     9$                      ;BRANCH IF GOOD
13126 063330  104003                          ERROR   .+3             ;FPP ERROR
13127                                                                 ;BAD ACO
13128                         ;UNDERFLOW TRAPS ENABLED
13129 063332  012702  002200          9$:     MOV     #2200,R2                ;SETUP FLOATING POINT STATUS
13130 063336  170102                          LDFPS   R2              ;LOAD FPS
13131 063340  012737  063366  000244          MOV     #11$,@#FPVEC    ;REPOSITION TRAP VECTOR
13132 063346  012704  003334                  MOV     #TAB7,R4                ;POINT TO FSRC DATA
13133 063352  012701  003714                  MOV     #TAB41,R1               ;POINT TO ACO DATA
13134 063356  172411                          LDD     (R1),AC0                ;LOAD ACO WITH TEST DATA
13135 063360  172014                          ADDD    (R4),AC0        ;*TEST INSTRUCTION
13136 063362  170000          10$:    CFCC                    ;COPTY FPP CC
13137 063364  104003                          ERROR   .+3             ;FPP ERROR
13138                                                                 ;FAILED TO TRAP ON UNDERFLOW
13139 063366  170203          11$:    STFPS   R3              ;SAVE FPS
13140 063370  012701  003142                  MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
13141 063374  174011                          STD     AC0,(R1)        ;SAVE ACO  RESULT
13142 063376  022703  102210                  CMP     #102210,R3              ;VERIFY STATUS
13143 063402  001401                          BEQ     12$                     ;BRANCH IF GOOD
13144 063404  104003                          ERROR   .+3             ;FPP ERROR
13145                                                                 ;BAD FPS
13146 063406  012605          12$:    MOV     (SP)+,R5                ;GET ERROR PC
13147 063410  020527  063362                  CMP     R5,#10$         ;VERIFY ERROR ADDRESS ON STACK
13148 063414  001401                          BEQ     13$                     ;BRANCH IF GOOD
13149 063416  104003                          ERROR   .+3             ;FPP ERROR
13150                                                                 ;BAD ERROR RETURN ON STACK
13151 063420  005726          13$:    TST     (SP)+                   ;RESTORE STACK
13152 063422  012704  003304                  MOV     #TAB5A,R4               ;POINT TO EXPECTED DATA
13153 063426  004767  051762                  JSR     R7,DATVER               ;VERIFY DATA
13154 063432  005767  117462                  TST     COUNT
13155 063436  001401                          BEQ     14$                     ;BRANCH IF GOOD
13156 063440  104003                          ERROR   .+3             ;FPP ERROR
13157                                                                 ;BAD ACO
13158                         ;UNDERFLOW WITH TRAPS DISBLED - NON-ZERO RESULT
13159 063442  012702  000200          14$:    MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13160 063446  170102                          LDFPS   R2                      ;LOAD FPS
13161 063450  012737  135360  000244          MOV     #WLDTRP,@#FPVEC         ;RESTORE TRAP VECTOR
13162 063456  012704  003714                  MOV     #TAB41,R4               ;POINT TO FSRC DATA
13163 063462  012701  003724                  MOV     #TAB42,R1               ;POINT TO ACO DATA
13164 063466  172411                          LDD     (R1),AC0                ;LOAD ACO WITH TEST DATA
13165 063470  172014                          ADDD    (R4),AC0        ;*TEST INSTRUCTION
13166 063472  170203                          STFPS   R3              ;SAVE FPS
13167 063474  012701  003142                  MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
13168 063500  174011                          STD     AC0,(R1)        ;SAVE ACO  RESULT
13169 063502  022703  000204                  CMP     #204,R3         ;VERIFY STATUS
13170 063506  001401                          BEQ     15$                     ;BRANCH IF GOOD
13171 063510  104003                          ERROR   .+3             ;FPP ERROR
13172                                                                 ;BAD FPS
13173 063512  012704  003314          15$:    MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
13174 063516  004767  051672                  JSR     R7,DATVER               ;VERIFY DATA
13175 063522  005767  117372                  TST     COUNT
13176 063526  001401                          BEQ     16$                     ;BRANCH IF GOOD
13177 063530  104003                          ERROR   .+3             ;FPP ERROR
```

```
13178                                                           ;BAD ACO
13179                                   ;UNERFLOW WITH TRAPS ENABLED - NON-ZERO  RESULT
13180 063532  012702  102200            16$:    MOV     #102200,R2              ;SETUP FLOATING POINT STATUS
13181 063536  170102                            LDFPS   R2                      ;LOAD FPS
13182 063540  012737  063566  000244            MOV     #18$,@#FPVEC            ;RESTORE TRAP VECTOR
13183 063546  012704  003714                    MOV     #TAB41,R4               ;POINT TO FSRC DATA
13184 063552  012701  003724                    MOV     #TAB42,R1               ;POINT TO ACO DATA
13185 063556  172411                            LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13186 063560  172014                            ADDD    (R4),AC0                ;*TEST INSTRUCTION
13187 063562  170000                    17$:    CFCC
13188 063564  104003                            ERROR   .+3                     ;FPP ERROR
13189                                                                           ;NO TRAP ON UNDERFLOW
13190 063566  170203                    18$:    STFPS   R3                      ;SAVE FPS
13191 063570  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13192 063574  174011                            STD     AC0,(R1)                ;SAVE AC0  RESULT
13193 063576  012600                            MOV     (SP)+,R0                ;SAVE STACK CONTENTS
13194 063600  005726                            TST     (SP)+                   ;CLEAN UP STACK
13195 063602  022700  063562                    CMP     #17$,R0                 ;VERIFY RETURN ADDRESS
13196 063606  001401                            BEQ     19$                     ;BRANCH IF GOOD
13197 063610  104003                            ERROR   .+3                     ;FPP ERROR
13198                                                                           ;BAD RETURN ADDRESS
13199 063612  022703  102204            19$:    CMP     #102204,R3              ;VERIFY STATUS
13200 063616  001401                            BEQ     20$                     ;BRANCH IF GOOD
13201 063620  104003                            ERROR   .+3                     ;FPP ERROR
13202                                                                           ;BAD FPS
13203 063622  012704  003734            20$:    MOV     #TAB43,R4               ;POINT TO EXPECTED DATA
13204 063626  004767  051562                    JSR     R7,DATVER               ;VERIFY DATA
13205 063632  005767  117262                    TST     COUNT
13206 063636  001401                            BEQ     21$                     ;BRANCH IF GOOD
13207 063640  104003                            ERROR   .+3                     ;FPP ERROR
13208                                                                           ;BAD AC0
13209 063642                            21$:
13210
13211
13214                                   ;
13215                                   ;
13216                                   ;-----------------------------------
13217                                   ;TEST LDCFD, LDCDF
13218                                   ;
13219                                           ;
13220                                           ;
13221                                           ;
13222 063642                            MLDC:
13223
13224                                   ;
13225                                   ;TRUNCATE
13226 063642  012702  000300                    MOV     #300,R2                 ;SETUP FLOATING POINT STATUS
13227 063646  170102                            LDFPS   R2                      ;LOAD FPS
13228 063650  012704  003744                    MOV     #TAB45,R4               ;POINT TO FSRC DATA
13229 063654  012701  003314                    MOV     #TAB6,R1                ;POINT TO ACO DATA
13230 063660  172411                            LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13231 063662  177424                            LDCFD   (R4)+,AC0               ;*TEST INSTRUCTION
13232 063664  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13233 063670  174011                            STD     AC0,(R1)                ;SAVE AC0  RESULT
13234 063672  022704  003750                    CMP     #TAB45+4,R4             ;VERIFY AUTO-INC
13235 063676  001401                            BEQ     1$                      ;BRANCH IF GOOD AUTO-INC
13236 063700  104003                            ERROR   .+3                     ;FPP ERROR
```

FLOATING POINT TESTS

```
13237
13238 063702  012704  003754          1$:     MOV     @TAB46,R4               ;BAD AUTO-INC
13239 063706  004767  051502                  JSR     R7,DATVER               ;POINT TO EXPECTED DATA
13240 063712  005767  117202                  TST     COUNT                   ;VERIFY DATA
13241 063716  001401                          BEQ     2$                      ;BRANCH IF GOOD
13242 063720  104003                          ERROR   .3                      ;FPP ERROR
13243                                                                         ;BAD ACO
13244                                  ;AUTO-INC DOUBLE MODE
13245 063722  005002                  2$:     CLR     R2                      ;SETUP FLOATING POINT STATUS
13246 063724  170102                          LDFPS   R2                      ;LOAD FPS
13247 063726  012704  003744                    MOV     @TAB45,R4               ;POINT TO FSRC DATA
13248 063732  012701  003464                    MOV     @TAB16,R1               ;POINT TO ACO DATA
13249 063736  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13250 063740  177424                          LDCDF   (R4)+,AC0               ;*TEST INSTRUCTION
13251 063742  020427  003754                  CMP     R4,@TAB45+10            ;VERIFY AUTO-INC
13252 063746  001401                          BEQ     3$                      ;BRANCH IF GOOD
13253 063750  104003                          ERROR   .3                      ;FPP ERROR
13254                                                                         ;BAD AUTO-INC ON DOUBLE
13255 063752  170203                  3$:     STFPS   R3                      ;SAVE FPS
13256 063754  012701  003142                  MOV     @RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13257 063760  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13258 063762  022703  000000                  CMP     @0,R3                   ;VERIFY STATUS
13259 063766  001401                          BEQ     4$                      ;BRANCH IF GOOD
13260 063770  104003                          ERROR   .3                      ;FPP ERROR
13261                                                                         ;BAD FPS
13262 063772  012704  004014          4$:     MOV     @TAB49,R4               ;POINT TO EXPECTED DATA
13263 063776  004767  051412                  JSR     R7,DATVER               ;VERIFY DATA
13264 064002  005767  117112                  TST     COUNT
13265 064006  001401                          BEQ     5$                      ;BRANCH IF GOOD
13266 064010  104003                          ERROR   .3                      ;FPP ERROR
13267                                                                         ;BAD AC0
13268                                  ;LDCFD GR7
13269 064012  012702  000200          5$:     MOV     @200,R2                 ;SETUP FLOATING POINT STATUS
13270 064016  170102                          LDFPS   R2                      ;LOAD FPS
13271 064020  005003                          CLR     R3
13272 064022  177427  043243                  LDCFD   @5203,AC0               ;*TEST INSTRUCTION
13273 064026  005203                          INC     R3
13274 064030  005203                          INC     R3
13275 064032  005203                          INC     R3                      ;IF LDCFD WORKED, R3 SHOULD=3
13276 064034  022703  000003                  CMP     @3,R3                   ;VERIFY CORRECT PROGRAM FLOW
13277 064040  001401                          BEQ     6$                      ;BRANCH IF GOOD
13278 064042  104003                          ERROR   .3                      ;FPP ERROR
13279                                                                         ;BAD PROGRAM FLOW
13280                                  ;NEGATIVE OPERANDS
13281 064044  012702  000200          6$:     MOV     @200,R2                 ;SETUP FLOATING POINT STATUS
13282 064050  170102                          LDFPS   R2                      ;LOAD FPS
13283 064052  012704  003764                    MOV     @TAB47,R4               ;POINT TO FSRC DATA
13284 064056  012701  003744                    MOV     @TAB45,R1               ;POINT TO ACO DATA
13285 064062  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13286 064064  177414                          LDCFD   (R4),AC0                ;*TEST INSTRUCTION
13287 064066  170203                          STFPS   R3                      ;SAVE FPS
13288 064070  012701  003142                  MOV     @RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13289 064074  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13290 064076  022703  000210                  CMP     @210,R3                 ;VERIFY STATUS
13291 064102  001401                          BEQ     7$                      ;BRANCH IF GOOD
13292 064104  104003                          ERROR   .3                      ;FPP ERROR
13293                                                                         ;BAD FPS
```

FLOATING POINT TESTS

```
13294 064106  012704  004004          7$:     MOV     #TAB48,R4                       ;POINT TO EXPECTED DATA
13295 064112  004767  051276                  JSR     R7,DATVER                       ;VERIFY DATA
13296 064116  005767  116776                  TST     COUNT
13297 064122  001401                          BEQ     8$                      ;BRANCH IF GOOD
13298 064124  104003                          ERROR   .3              ;FPP ERROR
13299                                                                 ;BAD ACO
13300                                         ;LOAD A ZERO
13301 064126  012702  000200          8$:     MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13302 064132  170102                          LDFPS   R2                      ;LOAD FPS
13303 064134  012704  003314                  MOV     #TAB6,R4                        ;POINT TO FSRC DATA
13304 064140  012701  004004                  MOV     #TAB48,R1                       ;POINT TO ACO DATA
13305 064144  172411                          LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13306 064146  177414                          LDCFD   (R4),AC0                        ;*TEST INSTRUCTION
13307 064150  170203                          STFPS   R3                      ;SAVE FPS
13308 064152  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13309 064156  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13310 064160  022703  000204                  CMP     #204,R3                 ;VERIFY STATUS
13311 064164  001401                          BEQ     9$                      ;BRANCH IF GOOD
13312 064166  104003                          ERROR   .3              ;FPP ERROR
13313                                                                 ;BAD FPS
13314 064170  012704  003314          9$:     MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13315 064174  004767  051214                  JSR     R7,DATVER                       ;VERIFY DATA
13316 064200  005767  116714                  TST     COUNT
13317 064204  001401                          BEQ     10$                     ;BRANCH IF GOOD
13318 064206  104003                          ERROR   .3              ;FPP ERROR
13319                                                                 ;BAD AC0
13320 064210                          10$:
13321
13322                                         ;
13325                                         ;
13326                                         ;
13327                                         ;-------------------------------
13328                                         ;TEST CMPD
13329                                         ;
13330                                                 ;
13331                                                 ;
13332                                                 ;
13333 064210                          MCMPD:
13334
13335                                         ;
13336                                         ;  CMPD WITH FRSC=AC0=0
13337 064210  005037  003030                          CLR     @#FLAG                  ;SIGNAL THAT AC0 REMAINS CONSTANT
13338 064214  004767  000152                          JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13339 064220  000000  000000  000000                  .WORD   0,0,0,0                 ;AC0 AT START
      064226  000000
13340 064230  000000  000000  000000                  .WORD   0,0,0,0                 ;FSRC AT START
      064236  000000
13341 064240  000200                                  .WORD   200                     ;FPS AT START (D)
13342 064242  000204                                  .WORD   204                     ;FPS AT END
13343                                           ;  CMPD WITH EXP[FSRC]=0, EXP[AC0]=0
13344 064244  012737  000001  003030                  MOV     #1,@#FLAG               ;SIGNAL THAT AC0 WILL = 0
13345 064252  004767  000114                          JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13346 064256  000000  000000  000000                  .WORD   0,0,0,125252            ;AC0 AT START
      064264  125252
13347 064266  000100  000022  000123                  .WORD   100,22,123,123          ;FSRC AT START
      064274  000123
13348 064276  000200                                  .WORD   200                     ;FPS AT START (D)
```

FLOATING POINT TESTS

```
13349 064300  000204                        .WORD   204                     ;FPS AT END
13350                               ;  CMPD FSRC>EXP[AC0]=0
13351 064302  005037  003030                CLR     @#FLAG                  ;AC0 REMAINS UNCHANGED
13352 064306  004767  000060                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13353 064312  000400  012346  012346        .WORD   400,12346,12346,23      ;AC0 AT START
      064320  000023
13354 064322  000200  000000  000000        .WORD   200,0,0,0               ;FSRC AT START
      064330  000000
13355 064332  000200                        .WORD   200                     ;FPS AT START (D)
13356 064334  000210                        .WORD   210                     ;FPS AT END
13357                               ; CMPD FSRC=AC0>0
13358 064336  004767  000030                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13359 064342  077777  177777  177777        .WORD   77777,-1,-1,-1          ;AC0 AT START
      064350  177777
13360 064352  077777  177777  177777        .WORD   77777,-1,-1,-1          ;FSRC AT START
      064360  177777
13361 064362  000200                        .WORD   200                     ;FPS AT START (D)
13362 064364  000204                        .WORD   204                     ;FPS AT END
13363 064366  000167  000116                JMP     @OP44                   ;HOP OVER SUBROUTINE
13364
13365                               ;*X*X*X
```

FLOATING POINT TESTS

```
13402 064470  004767  050720          4$:     JSR     R7,DATVER                       ;VERIFY DATA
13403 064474  005767  116420                  TST     COUNT
13404 064500  001401                          BEQ     5$                      ;BRANCH IF GOOD
13405 064502  104003                          ERROR   .+3             ;FPP ERROR
13406                                                                 ;BAD ACO
13407 064504  000165  000024          5$:     JMP     24(R5)                  ;RETURN
13408 064510                          MOP44:
13409
13410
13411
13414                                 ;
13415                                 ;
13416                                 ;---------------------------------
13417                                 ;TEST DIVF
13418                                 ;
13419                                         ;
13420                                         ;
13421                                         ;
13422 064510                          MDIVF:
13423
13424                                 ;
13425                                 ;1/EXP[AC]=FSRC=0
13426 064510  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT, BUT FEC
13427 064516  004767  000706                  JSR     R7,DVFSUB               ;DO TEST
13428 064522  000100  000027                  .WORD   100,27          ;AC0
13429 064526  000000  000000                  .WORD   0,0             ;FSRC
13430 064532  000100  000027                  .WORD   100,27          ;RESULT
13431 064536  040000                          .WORD   40000           ; TEST FPS
13432 064540  140000                          .WORD   140000          ;RESULT FPS
13433 064542  000004                          .WORD   4               ;FEC
13434                                 ;2/AC=EXP[FSRC]=0
13435                                 ;TRAPS ENABLED
13436 064544  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13437 064552  004767  000652                  JSR     R7,DVFSUB               ;DO TEST
13438 064556  000000  000000                  .WORD   0,0             ;AC0
13439 064562  000100  000000                  .WORD   100,0           ;FSRC
13440 064566  000000  000000                  .WORD   0,0             ;RESULT
13441 064572  000000                          .WORD   0               ; TEST FPS
13442 064574  100000                          .WORD   100000          ;RESULT FPS
13443 064576  000004                          .WORD   4               ;FEC
13444                                 ;3/FSRC>AC0=0
13445 064600  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13446 064604  004767  000620                  JSR     R7,DVFSUB               ;DO TEST
13447 064610  000177  000234                  .WORD   177,234         ;AC0
13448 064614  004100  000000                  .WORD   4100,0          ;FSRC
13449 064620  000000  000000                  .WORD   0,0             ;RESULT
13450 064624  007400                          .WORD   7400            ; TEST FPS
13451 064626  007404                          .WORD   7404            ;RESULT FPS
13452                                 ;4/AC0>EXP[FSRC]=0
13453 064630  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13454 064636  004767  000566                  JSR     R7,DVFSUB               ;DO TEST
13455 064642  040200  104210                  .WORD   40200,104210    ;AC0
13456 064646  000125  025252                  .WORD   125,25252       ;FSRC
13457 064652  040200  104210                  .WORD   40200,104210    ;RESULT
13458 064656  007557                          .WORD   7557            ; TEST FPS
13459 064660  107557                          .WORD   107557          ;RESULT FPS
13460 064662  000004                          .WORD   4               ;FEC
```

F3

FLOATING POINT TESTS

```
13461
13462 064664  005037  003030                ;5/EXP[AC]=EXP[FSRC]
                            CLR     @#FLAG          ;NO INTERRUPT
13463 064670  004767  000534                JSR     R7,DVFSUB               ;DO TEST
13464 064674  077760  177777                .WORD   77760,-1                ;AC0
13465 064700  077760  000000                .WORD   77760,0         ;FSRC
13466 064704  040200  104210                .WORD   40200,104210    ;RESULT
13467 064710  007414                        .WORD   7414            ; TEST FPS
13468 064712  007400                        .WORD   7400            ;RESULT FPS
13469                               ;6/AC=FSRC
13470 064714  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13471 064720  004767  000504                JSR     R7,DVFSUB               ;DO TEST
13472 064724  052525  052525                .WORD   52525,52525             ;AC0
13473 064730  052525  052525                .WORD   52525,52525             ;FSRC
13474 064734  040200  000000                .WORD   40200,0         ;RESULT
13475 064740  007400                        .WORD   7400            ; TEST FPS
13476 064742  007400                        .WORD   7400            ;RESULT FPS
13477                               ;7/FSRC>0<AC0, ROUND
13478 064744  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13479 064750  004767  000454                JSR     R7,DVFSUB               ;DO TEST
13480 064754  077777  125252                .WORD   77777,125252    ;AC0
13481 064760  040300  000000                .WORD   40300,0         ;FSRC
13482 064764  077652  070707                .WORD   77652,070707                    ;RESULT
13483 064770  007400                        .WORD   7400            ; TEST FPS
13484 064772  007400                        .WORD   7400            ;RESULT FPS
13485                               ;8/AC>0<FSRC
13486 064774  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13487 065000  004767  000424                JSR     R7,DVFSUB               ;DO TEST
13488 065004  055377  177777                .WORD   55377,-1                ;AC0
13489 065010  055300  000000                .WORD   55300,0         ;FSRC
13490 065014  040252  125252                .WORD   40252,125252    ;RESULT
13491 065020  000000                        .WORD   0               ; TEST FPS
13492 065022  000000                        .WORD   0               ;RESULT FPS
13493                               ;9/FSRC>AC>0
13494 065024  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13495 065030  004767  000374                JSR     R7,DVFSUB               ;DO TEST
13496 065034  064600  000001                .WORD   64600,1         ;AC0
13497 065040  066600  000000                .WORD   66600,0         ;FSRC
13498 065044  036200  000001                .WORD   36200,1         ;RESULT
13499 065050  000000                        .WORD   0               ; TEST FPS
13500 065052  000000                        .WORD   0               ;RESULT FPS
13501                               ;10/AC>FSRC>0
13502 065054  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13503 065060  004767  000344                JSR     R7,DVFSUB               ;DO TEST
13504 065064  012345  156024                .WORD   12345,156024    ;AC0
13505 065070  005600  000000                .WORD   05600,0         ;FSRC
13506 065074  044745  156024                .WORD   44745,156024                    ;RESULT
13507 065100  000017                        .WORD   17              ; TEST FPS
13508 065102  000000                        .WORD   0               ;RESULT FPS
13509                               ;11/FSRC<0
13510 065104  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13511 065110  004767  000314                JSR     R7,DVFSUB               ;DO TEST
13512 065114  040422  101010                .WORD   40422,101010    ;AC0
13513 065120  140511  101010                .WORD   140511,101010   ;FSRC
13514 065124  140072  020167                .WORD   140072,20167                    ;RESULT
13515 065130  000057                        .WORD   57              ; TEST FPS
13516 065132  000050                        .WORD   50              ;RESULT FPS
13517                               ;12/AC<0
```

FLOATING POINT TESTS

```
13518 065134  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13519 065140  004767  000264                  JSR     R7,DVFSUB               ;DO TEST
13520 065144  160077  000101                  .WORD   160077,101      ;ACO
13521 065150  040417  177777                  .WORD   40417,-1        ;FSRC
13522 065154  157651  143527                  .WORD   157651,143527           ;RESULT
13523 065160  000007                          .WORD   7               ; TEST FPS
13524 065162  000010                          .WORD   10              ;RESULT FPS
13525                               ;13/TRUNCATE TEST
13526 065164  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13527 065170  004767  000234                  JSR     R7,DVFSUB               ;DO TEST
13528 065174  060100  000177                  .WORD   60100,177       ;ACO
13529 065200  040300  000000                  .WORD   40300,0         ;FSRC
13530 065204  060000  000124                  .WORD   60000,124               ;RESULT
13531 065210  000040                          .WORD   40              ; TEST FPS
13532 065212  000040                          .WORD   40              ;RESULT FPS
13533                               ;14/ROUND TEST
13534
13535 065214  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13536 065220  004767  000204                  JSR     R7,DVFSUB               ;DO TEST
13537 065224  060100  000177                  .WORD   60100,177       ;ACO
13538 065230  040300  000000                  .WORD   40300,0 ;FSRC
13539 065234  060000  000125                  .WORD   60000,125               ;RESULT
13540 065240  000000                          .WORD   0               ; TEST FPS
13541 065242  000000                          .WORD   0               ;RESULT FPS
13542                               ;15/OVERFLOW, INTERRUPTS ENABLED
13543 065244  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13544 065252  004767  000152                  JSR     R7,DVFSUB               ;DO TEST
13545 065256  177700  000000                  .WORD   177700,0                ;ACO
13546 065262  000200  000000                  .WORD   200,0           ;FSRC
13547 065266  137700  000000                  .WORD   137700,0                ;RESULT
13548 065272  001100                          .WORD   1100            ; TEST FPS
13549 065274  101112                          .WORD   101112          ;RESULT FPS
13550 065276  000010                          .WORD   10              ;FEC
13551                               ;16/OVERFLOW, TRAPS DISABLED
13552 065300  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT
13553 065306  004767  000116                  JSR     R7,DVFSUB               ;DO TEST
13554 065312  000200  000000                  .WORD   200,0           ;ACO
13555 065316  177700  000000                  .WORD   177700,0                ;FSRC
13556 065322  000000  000000                  .WORD   0,0             ;RESULT
13557 065326  041100                          .WORD   41100           ; TEST FPS
13558 065330  041104                          .WORD   41104           ;RESULT FPS
13559 065332  000010                          .WORD   10              ;FEC OVERFLOW
13560                               ;17/UNDERFLOW, TRAPS ENABLED, UV RESULT
13561 065334  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13562 065342  004767  000062                  JSR     R7,DVFSUB               ;DO TEST
13563 065346  100200  000000                  .WORD   100200,0                ;ACO
13564 065352  040377  177777                  .WORD   40377,-1                ;FSRC
13565 065356  100000  000001                  .WORD   100000,1                ;RESULT
13566 065362  002000                          .WORD   2000            ; TEST FPS
13567 065364  102014                          .WORD   102014          ;RESULT FPS
13568 065366  000012                          .WORD   12              ;FEC
13569                               ;18/UNDERFLOW,TRAPS ENABLED, ROUND
13570 065370  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13571 065376  004767  000026                  JSR     R7,DVFSUB               ;DO TEST
13572 065402  030325  025252                  .WORD   30325,25252             ;ACO
13573 065406  076777  023456                  .WORD   76777,23456             ;FSRC
13574 065412  071525  157716                  .WORD   71525,157716            ;RESULT
```

FLOATING POINT TESTS

```
13575 065416  002537                        .WORD   2537            ; TEST FPS
13576 065420  102500                        .WORD   102500          ;RESULT FPS
13577 065422  000012                        .WORD   12              ;FEC
13578                                ;
13579
13580 065424  000167  000212                JMP     HOP10           ;GO TO NEXT TEST
13581                                ;*X*X*X
```

FLOATING POINT TESTS

```
13632 065556  026503  000016                CMP     16(R5),R3               ;VERIFY STATUS
13633 065562  001401                        BEQ     3$                      ;BRANCH IF GOOD
13634 065564  104003                        ERROR   .+3                 ;FPP ERROR
13635                                                                   ;BAD FPS
13636 065566  010504                3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13637 065570  062704  000010                ADD     #10,R4
13638 065574  004767  047576        4$:     JSR     R7,DATVFR                       ;VERIFY DATA
13639 065600  005767  115314                TST     COUNT
13640 065604  001401                        BEQ     5$                      ;BRANCH IF GOOD
13641 065606  104003                        ERROR   .+3                 ;FPP ERROR
13642                                                                   ;BAD ACO
13643 065610  005737  003030        5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13644 065614  001002                        BNE     6$                      ;BRANCH IF NEED TO CHECK
13645 065616  000165  000020                JMP     20(R5)                  ;RETURN FROM TEST
13646 065622  170301                6$:     STST    R1                      ;SAVE FEC
13647 065624  016504  000020                MOV     20(R5),R4               ;GET FEC
13648 065630  020401                        CMP     R4,R1                   ;VERIFY FEC
13649 065632  001401                        BEQ     7$                      ;BRANCH IF GOOD
13650 065634  104003                        ERROR   .+3                 ;FPP ERROR
13651                                                                   ;BAD FEC
13652 065636  000165  000022        7$:     JMP     22(R5)                  ;RETURN FROM TEST
13653                                     ;
13654 065642                              MOP10:
13657                                     ;
13658                                     ;
13659                                     ;---------------------------------
13660                                     ;TEST DIVD -
13661                                     ;
13662                                             ;
13663                                             ;
13664                                             ;
13665 065642                              MDIVD:
13666
13667                                     ;
13668                                     ;1/AC=FSRC=0 TRAPS DISABLED
13669 065642  012737  000002  003030              MOV     #2,@#FLAG                       ;NO INTERRUPT
13670 065650  004767  000516                      JSR     R7,DVDSUB                       ;DO TEST
13671 065654  000000  000000  000000              .WORD   0,0,0,1             ;AC0
      065662  000001
13672 065664  000100  000000  000000              .WORD   100,0,0,0           ;FSRC
      065672  000000
13673 065674  000000  000000  000000              .WORD   0,0,0,1             ;RESULT
      065702  000001
13674 065704  040000                            .WORD   40000               ; TEST FPS
13675 065706  140000                            .WORD   140000              ;RESULT FPS
13676 065710  000004                            .WORD   4                   ;FEC
13677                                     ;2/FSRC=0, TRAPS ENABLED
13678 065712  012737  000001  003030              MOV     #1,@#FLAG                       ;INTERRUPT
13679 065720  004767  000446                      JSR     R7,DVDSUB                       ;DO TEST
13680 065724  000402  000000  000000              .WORD   402,0,0,0           ;AC0
      065732  000000
13681 065734  000000  000000  000000              .WORD   0,0,0,0             ;FSRC
      065742  000000
13682 065744  000402  000000  000000              .WORD   402,0,0,0                       ;RESULT
      065752  000000
13683 065754  000200                            .WORD   200                 ; TEST FPS
13684 065756  100200                            .WORD   100200              ;RESULT FPS
```

J3

FLOATING POINT TESTS

```
13685 065760  000004                        .WORD   4                       ;FEC
13686                               ;3/ROUND
13687 065762  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
13688 065766  004767  000400                  JSR     R7,DVDSUB                       ;DO TEST
13689 065772  034300  000000  000000          .WORD   34300,0,0,1             ;AC0
      066000  000001
13690 066002  140300  000000  000000          .WORD   140300,0,0,0            ;FSRC
      066010  000000
13691 066012  134200  000000  000000          .WORD   134200,0,0,1            ;RESULT
      066020  000001
13692 066022  000200                        .WORD   200                     ; TEST FPS
13693 066024  000210                        .WORD   210                     ;RESULT FPS
13694                               ;4/TRUNCATE
13695 066026  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
13696 066032  004767  000334                  JSR     R7,DVDSUB                       ;DO TEST
13697 066036  034300  000000  000000          .WORD   34300,0,0,1             ;AC0
      066044  000001
13698 066046  140300  000000  000000          .WORD   140300,0,0,0            ;FSRC
      066054  000000
13699 066056  134200  000000  000000          .WORD   134200,0,0,0            ;RESULT
      066064  000000
13700 066066  000240                        .WORD   240                     ; TEST FPS
13701 066070  000250                        .WORD   250                     ;RESULT FPS
13702                               ;5/ROUND NEGATIVE AC, FSRC
13703 066072  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
13704 066076  004767  000270                  JSR     R7,DVDSUB                       ;DO TEST
13705 066102  177642  000000  000000          .WORD   177642,0,0,151          ;AC0
      066110  000151
13706 066112  166600  000000  000000          .WORD   166600,0,0,123          ;FSRC
      066120  000123
13707 066122  051242  000000  000000          .WORD   51242,0,0,0                     ;RESULT
      066130  000000
13708 066132  000200                        .WORD   200                     ; TEST FPS
13709 066134  000200                        .WORD   200                     ;RESULT FPS
13710                               ;6/TRUNCATE NEAGTIVE AC, FSRC
13711 066136  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
13712 066142  004767  000224                  JSR     R7,DVDSUB                       ;DO TEST
13713 066146  177642  000000  000000          .WORD   177642,0,0,151          ;AC0
      066154  000151
13714 066156  166600  000000  000000          .WORD   166600,0,0,123          ;FSRC
      066164  000123
13715 066166  051241  177777  177777          .WORD   51241,-1,-1,-1                  ;RESULT
      066174  177777
13716 066176  000240                        .WORD   240                     ; TEST FPS
13717 066200  000240                        .WORD   240                     ;RESULT FPS
13718                               ;7/AC=FSRC
13719 066202  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
13720 066206  004767  000160                  JSR     R7,DVDSUB                       ;DO TEST
13721 066212  055521  047621  100333          .WORD   55521,47621,100333,-1   ;AC0
      066220  177777
13722 066222  055521  047621  100333          .WORD   55521,47621,100333,-1   ;FSRC
      066230  177777
13723 066232  040200  000000  000000          .WORD   40200,0,0,0                     ;RESULT
      066240  000000
13724 066242  007717                        .WORD   7717                    ; TEST FPS
13725 066244  007700                        .WORD   7700                    ;RESULT FPS
13726                               ;8/UNDERFLOW TRAPS ENABLED, UV RESULT
```

K3

FLOATING POINT TESTS

```
13727 066246  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13728 066254  004767  000112                  JSR     R7,DVDSUB               ;DO TEST
13729 066260  100200  000000  000000          .WORD   100200,0,0,0            ;AC0
      066266  000000
13730 066270  077777  000000  000000          .WORD   77777,0,0,0             ;FSRC
      066276  000000
13731 066300  140400  100200  100200          .WORD   140400,100200,100200,100201     ;RESULT
      066306  100201
13732 066310  002200                          .WORD   2200            ; TEST FPS
13733 066312  102210                          .WORD   102210          ;RESULT FPS
13734 066314  000012                          .WORD   12              ;FEC
13735                       ;9/OVERFLOW TRAPS ENABLED
13736 066316  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13737 066324  004767  000042                  JSR     R7,DVDSUB               ;DO TEST
13738 066330  077000  123465  012346          .WORD   77000,123465,12346,525          ;AC0
      066336  000525
13739 066340  000303  000001  140000          .WORD   303,1,140000,140001             ;FSRC
      066346  140001
13740 066350  036650  163002  103645          .WORD   36650,163002,103645,64003       ;RESULT
      066356  064003
13741 066360  001700                          .WORD   1700            ; TEST FPS
13742 066362  101702                          .WORD   101702          ;RESULT FPS
13743 066364  000010                          .WORD   10              ;FEC
13744                       ;
13745
13746 066366  000167  000212                  JMP     HOP11                   ;HOP OVER SUBROUTINE
13747                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13748                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13749                       ;
13750                       ;DIVD SUBROUTINE:
13751                       ;               AC0
13752                       ;               FSRC
13753                       ;               FPS BEFORE EXECUTION
13754                       ;               FPS AFTER EXECUTION
13755                       ;               (FEC)
13756                       ;
13757                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13758                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13759                       ;
13760 066372  012605                  DVDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13761 066374  012737  066450  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
13762 066402  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13763 066406  170102                          LDFPS   R2                      ;LOAD FPS
13764 066410  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13765 066412  062704  000010                  ADD     #10,R4
13766 066416  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13767 066420  016502  000030                  MOV     30(R5),R2               ;GET TEST FPS
13768 066424  170102                          LDFPS   R2                      ;LOAD TEST FPS
13769                       ;
13770 066426  174414                          DIVD    (R4),AC0                ;*TEST INSTRUCTION
13771 066430  170000                  1$:     CFCC                            ;WAIT FOR POSSIBLE FPA TRAP.
13772                       ;
13773                       ;INSTRUCTION DIDNT TRAP
13774 066432  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13775 066440  001420                          BEQ     2$                      ;BRANCH IF GOOD
13776 066442  104003                          ERROR   +3              ;FPP ERROR
13777                                                                 ;INSTRUCTION SHOULD HAVE TRAPPED
```

L3

FLOATING POINT TESTS

```
13778 066444  000167  000032              JMP     2$                      ;REJOIN CODE
13779                                     ;
13780                                     ;INSTRUCTION TRAPPED
13781 066450  032737  000001  003030  50$:  BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13782 066456  001003                      BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13783 066460  104003                      ERROR   .+3             ;FPP ERROR
13784                                                             ;INSTRUCTION WASNT SUPPOSE TO TRAP
13785 066462  000167  000014              JMP     2$                      ;REJOIN CODE
13786 066466  012604                51$:  MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
13787 066470  005726                      TST     (SP)+                   ;CLEAN UP STACK
13788 066472  022704  066430              CMP     #1$,R4                  ;
13789 066476  001401                      BEQ     2$                      ;BRANCH IF GOOD COMPARE
13790 066500  104003                      ERROR   .+3             ;FPP ERROR
13791                                                             ;PC WAS INCORRECT
13792                                     ;
13793                                     ;COMMON CODE FOR TRAP AND NO TRAP
13794 066502  170203                2$:   STFPS   R3                      ;SAVE FPS
13795 066504  012702  000200              MOV     #200,R2                 ;SET FPP TO DOUBLE
13796 066510  170102                      LDFPS   R2
13797 066512  012701  003142              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13798 066516  174011                      STD     AC0,(R1)                ;SAVE AC0  RESULT
13799 066520  026503  000032              CMP     32(R5),R3               ;VERIFY STATUS
13800 066524  001401                      BEQ     3$                      ;BRANCH IF GOOD
13801 066526  104003                      ERROR   .+3             ;FPP ERROR
13802                                                             ;BAD FPS
13803 066530  010504                3$:   MOV     R5,R4                   ;POINT TO EXPECTED DATA
13804 066532  062704  000020              ADD     #20,R4
13805 066536  004767  046652        4$:   JSR     R7,DATVER                       ;VERIFY DATA
13806 066542  005767  114352              TST     COUNT
13807 066546  001401                      BEQ     5$                      ;BRANCH IF GOOD
13808 066550  104003                      ERROR   .+3             ;FPP ERROR
13809                                                             ;BAD AC0
13810 066552  005737  003030        5$:   TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13811 066556  001002                      BNE     6$                      ;BRANCH IF NEED TO CHECK
13812 066560  000165  000034              JMP     34(R5)                  ;RETURN FROM TEST
13813 066564  170301                6$:   STST    R1                      ;SAVE FEC
13814 066566  016504  000034              MOV     34(R5),R4               ;GET FEC
13815 066572  020401                      CMP     R4,R1                   ;VERIFY FEC
13816 066574  001401                      BEQ     7$                      ;BRANCH IF GOOD
13817 066576  104003                      ERROR   .+3             ;FPP ERROR
13818                                                                     ;BAD FEC
13819 066600  000165  000036        7$:   JMP     36(R5)                  ;RETURN FROM TEST
13820                                     ;
13821 066604                        MOP11:
13824                                     ;
13825                                     ;
13826                                     ;---------------------------------
13827                                     ;TEST MULF
13828                                     ;
13829                                             ;
13830                                             ;
13831                                             ;
13832 066604                        MMULF:
13833
13834                                     ;
13835                                     ;1/AC0=FSRC=0 -INTERRUPTS DISABLED
13836 066604  005037  003030              CLR     @#FLAG                  ;NO INTERRUPT
```

M3

FLOATING POINT TESTS

```
13837 066610  004767  000564                JSR     R7,MLFSUB               ;DO TEST
13838 066614  000C00  000000                .WORD   0,0     ;AC0
13839 066620  000000  000000                .WORD   0,0     ;FSRC
13840 066624  000000  000000                .WORD   0,0                     ;RESULT
13841 066630  007517                        .WORD   7517                    ; TEST FPS
13842 066632  007504                        .WORD   7504                    ;RESULTANT FPS
13843                             ;2/AC>FSRC=0 - INTERRUPTS ON
13844 066634  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13845 066640  004767  000534                JSR     R7,MLFSUB                       ;DO TEST
13846 066644  000200  000000                .WORD   200,0   ;AC0
13847 066650  000000  000000                .WORD   0,0     ;FSRC
13848 066654  000000  000000                .WORD   0,0                     ;RESULT
13849 066660  000013                        .WORD   13                      ; TEST FPS
13850 066662  000004                        .WORD   4                       ;RESULTANT FPS
13851                             ;3/AC=0 FSRC>0 -
13852 066664  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13853 066670  004767  000504                JSR     R7,MLFSUB                       ;DO TEST
13854 066674  000100  000000                .WORD   100,0   ;AC0
13855 066700  000300  000000                .WORD   300,0   ;FSRC
13856 066704  000000  000000                .WORD   0,0                     ;RESULT
13857 066710  007500                        .WORD   7500                    ; TEST FPS
13858 066712  007504                        .WORD   7504                    ;RESULTANT FPS
13859                             ;4/AC=1 >FSRC - ROUND
13860 066714  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13861 066720  004767  000454                JSR     R7,MLFSUB                       ;DO TEST
13862 066724  040200  000000                .WORD   40200,0 ;AC0
13863 066730  040177  177777                .WORD   40177,-1                ;FSRC
13864 066734  040177  177777                .WORD   40177,-1                        ;RESULT
13865 066740  000000                        .WORD   0                       ; TEST FPS
13866 066742  000000                        .WORD   0                       ;RESULTANT FPS
13867                             ;5/TRUNCATE
13868 066744  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13869 066750  004767  000424                JSR     R7,MLFSUB                       ;DO TEST
13870 066754  040177  177777                .WORD   40177,-1                ;AC0
13871 066760  040200  000000                .WORD   40200,0 ;FSRC
13872 066764  040177  177777                .WORD   40177,-1                        ;RESULT
13873 066770  000040                        .WORD   40                      ; TEST FPS
13874 066772  000040                        .WORD   40                      ;RESULTANT FPS
13875                             ;6/NORMALIZE
13876 066774  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13877 067000  004767  000374                JSR     R7,MLFSUB                       ;DO TEST
13878 067004  040100  000000                .WORD   40100,0 ;AC0
13879 067010  040100  000000                .WORD   40100,0 ;FSRC
13880 067014  040020  000000                .WORD   40020,0                 ;RESULT
13881 067020  000012                        .WORD   12                      ; TEST FPS
13882 067022  000000                        .WORD   0                       ;RESULTANT FPS
13883                             ;7/ROUND
13884 067024  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13885 067030  004767  000344                JSR     R7,MLFSUB                       ;DO TEST
13886 067034  017500  000000                .WORD   17500,0 ;AC0
13887 067040  023652  125252                .WORD   23652,125252            ;FSRC
13888 067044  003177  177777                .WORD   3177,-1                 ;RESULT
13889 067050  007417                        .WORD   7417                    ; TEST FPS
13890 067052  007400                        .WORD   7400                    ;RESULTANT FPS
13891                             ;8/AC>0>FSRC ROUND
13892 067054  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
13893 067060  004767  000314                JSR     R7,MLFSUB                       ;DO TEST
```

FLOATING POINT TESTS

```
13894 067064  040342  177777                .WORD   40342,-1        ;ACO
13895 067070  176543  025252                .WORD   176543,025252   ;FSRC
13896 067074  176711  067324                .WORD   176711,67324            ;RESULT
13897 067100  007500                        .WORD   7500            ; TEST FPS
13898 067102  007510                        .WORD   7510            ;RESULTANT FPS
13899                               ;9/IAC<FSRC<0, ROUND
13900 067104  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13901 067110  004767  000264                JSR     R7,MLFSUB               ;DO TEST
13902 067114  144600  000000                .WORD   144600,0                ;ACO
13903 067120  154000  000000                .WORD   154000,0                ;FSRC
13904 067124  060400  000000                .WORD   60400,0         ;RESULT
13905 067130  000017                        .WORD   17              ; TEST FPS
13906 067132  000000                        .WORD   0               ;RESULT FPS
13907                               ;10/AC<FSRC, ROUND
13908 067134  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13909 067140  004767  000234                JSR     R7,MLFSUB               ;DO TEST
13910 067144  060000  000000                .WORD   60000,0         ;ACO
13911 067150  140377  177776                .WORD   140377,177776           ;FSRC
13912 067154  160177  177776                .WORD   160177,177776           ;RESULT
13913 067160  000017                        .WORD   17              ; TEST FPS
13914 067162  000010                        .WORD   10              ;RESULT FPS
13915                               ;11/AC>0>FSRC, TRUNCATE
13916 067164  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13917 067170  004767  000204                JSR     R7,MLFSUB               ;DO TEST
13918 067174  060000  000000                .WORD   60000,0         ;ACO
13919 067200  140377  177776                .WORD   140377,177776           ;FSRC
13920 067204  160177  177776                .WORD   160177,177776           ;RESULT
13921 067210  007547                        .WORD   7547            ; TEST FPS
13922 067212  007550                        .WORD   7550            ;RESULT FPS
13923                               ;12/UNDERFLOW, NO INTERRUPTS
13924 067214  012737  000002  003030        MOV     #2,@#FLAG               ;NO INTERRUPT
13925 067222  004767  000152                JSR     R7,MLFSUB               ;DO TEST
13926 067226  000200  000001                .WORD   200,1           ;ACO
13927 067232  000200  000001                .WORD   200,1           ;FSRC
13928 067236  040200  000002                .WORD   40200,2         ;RESULT
13929 067242  042117                        .WORD   42117           ; TEST FPS
13930 067244  142100                        .WORD   142100          ;RESULT FPS
13931 067246  000012                        .WORD   12              ;FEC
13932                               ;13/OVERFLOW, TRAP
13933 067250  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
13934 067256  004767  000116                JSR     R7,MLFSUB               ;DO TEST
13935 067262  177777  177777                .WORD   177777,-1               ;ACO
13936 067266  040300  000000                .WORD   40300,0         ;FSRC
13937 067272  100077  177777                .WORD   100077,-1               ;RESULT
13938 067276  001117                        .WORD   1117            ; TEST FPS
13939 067300  101116                        .WORD   101116          ;RESULT FPS
13940 067302  000010                        .WORD   10              ;FEC
13941                               ;14/OVERFLOW NO TRAP
13942 067304  012737  000002  003030        MOV     #2,@#FLAG               ;NO INTERRUPT
13943 067312  004767  000062                JSR     R7,MLFSUB               ;DO TEST
13944 067316  077700  000000                .WORD   77700,0         ;ACO
13945 067322  077700  000000                .WORD   77700,0         ;FSRC
13946 067326  000000  000000                .WORD   0,0             ;RESULT
13947 067332  040117                        .WORD   40117           ; TEST FPS
13948 067334  040106                        .WORD   40106           ;RESULT FPS
13949 067336  000010                        .WORD   10              ;FEC
13950                               ;15/UNDEFINED VARIABLE IN FSRC, TRAP ENABLED
```

FLOATING POINT TESTS

```
13951 067340  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13952 067346  004767  000026                  JSR     R7,MLFSUB               ;DO TEST
13953 067352  123465  000000                  .WORD   123465,0                ;AC0
13954 067356  100022  000000                  .WORD   100022,0                ;FSRC
13955 067362  123465  000000                  .WORD   123465,0                ;RESULT
13956 067366  004000                          .WORD   4000            ; TEST FPS
13957 067370  104000                          .WORD   104000          ;RESULT FPS
13958 067372  000014                          .WORD   14              ;FEC
13959                                 ;
13960
13961 067374  000167  000212                  JMP     MOP12
13962                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13963                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13964                                 ;
13965                                 ;               AC0
13966                                 ;               FSRC
13967                                 ;               FPS BEFORE EXECUTION
13968                                 ;               FPS AFTER EXECUTION
13969                                 ;               (FEC)
13970                                 ;
13971                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13972                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13973                                 ;
13974 067400  012605                  MLFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13975 067402  012737  067456  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
13976 067410  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13977 067414  170102                          LDFPS   R2                      ;LOAD FPS
13978 067416  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13979 067420  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13980 067422  062704  000004                  ADD     #4,R4
13981 067426  016502  000014                  MOV     14(R5),R2               ;GET TEST FPS
13982 067432  170102                          LDFPS   R2                      ;LOAD TEST FPS
13983                                 ;
13984 067434  171014                          MULF    (R4),AC0                ;*TEST INSTRUCTION
13985 067436  170001                  1$:     SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
13986                                 ;
13987                                 ;INSTRUCTION DIDNT TRAP
13988 067440  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13989 067446  001420                          BEQ     2$                      ;BRANCH IF GOOD
13990 067450  104003                          ERROR   .3              ;FPP ERROR
13991                                                                 ;INSTRUCTION SHOULD HAVE TRAPPED
13992 067452  000167  000032                  JMP     2$                      ;REJOIN CODE
13993                                 ;
13994                                 ;INSTRUCTION TRAPPED
13995 067456  032737  000001  003030  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13996 067464  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13997 067466  104003                          ERROR   .3              ;FPP ERROR
13998                                                                 ;INSTRUCTION WASNT SUPPOSE TO TRAP
13999 067470  000167  000014                  JMP     2$                      ;REJOIN CODE
14000 067474  012604                  51$:     MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14001 067476  005726                          TST     (SP)+                   ;CLEAN UP STACK
14002 067500  022704  067436                  CMP     #1$,R4                  ;
14003 067504  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
14004 067506  104003                          ERROR   .3              ;FPP ERROR
14005                                                                 ;PC WAS INCORRECT
14006                                 ;
14007                                 ;COMMON CODE FOR TRAP AND NO TRAP
```

FLOATING POINT TESTS

```
14008 067510  170203                  2$:     STFPS   R3                      ;SAVE FPS
14009 067512  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
14010 067516  170102                          LDFPS   R2
14011 067520  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14012 067524  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
14013 067526  026503  000016                  CMP     16(R5),R3               ;VERIFY STATUS
14014 067532  001401                          BEQ     3$                      ;BRANCH IF GOOD
14015 067534  104003                          ERROR   .3              ;FPP ERROR
14016                                                                 ;BAD FPS
14017 067536  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14018 067540  062704  000010                  ADD     #10,R4
14019 067544  004767  045626          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
14020 067550  005767  113344                  TST     COUNT
14021 067554  001401                          BEQ     5$                      ;BRANCH IF GOOD
14022 067556  104003                          ERROR   .3              ;FPP ERROR
14023                                                                 ;BAD AC0
14024 067560  005737  003030          5$:     TST     B#FLAG                  ;SEE IF NEED TO CHECK FEC
14025 067564  001002                          BNE     6$                      ;BRANCH IF NEED TO CHECK
14026 067566  000165  000020                  JMP     20(R5)                  ;RETURN FROM TEST
14027                                 ;VERIFY ERROR STATUS
14028 067572  170301                  6$:     STST    R1                      ;SAVE FEC
14029 067574  016504  000020                  MOV     20(R5),R4               ;GET FEC
14030 067600  020401                          CMP     R4,R1                   ;VERIFY FEC
14031 067602  001401                          BEQ     7$                      ;BRANCH IF GOOD
14032 067604  104003                          ERROR   .3              ;FPP ERROR
14033                                                                 ;BAD FEC
14034 067606  000165  000022          7$:     JMP     22(R5)          ;RETURN FROM TEST
14035 067612                          MOP12:
14038                                 ;
14039                                 ;
14040                                 ;---------------------------------
14041                                 ;TEST MULD
14042                                 ;
14043                                         ;
14044                                         ;
14045                                         ;
14046 067612                          MMULD:
14047
14048                                 ;
14049                                 ;1/AC=0
14050 067612  005037  003030                  CLR     B#FLAG                  ;NO INTERRUPT
14051 067616  004767  000554                  JSR     R7,MLDSUB                       ;DO TEST
14052 067622  000100  000000  000000          .WORD   100,0,0,0               ;AC0
      067630  000000
14053 067632  000411  177777  000000          .WORD   411,-1,0,1              ;FSRC
      067640  000001
14054 067642  000000  000000  000000          .WORD   0,0,0,0                 ;RESULT
      067650  000000
14055 067652  000200                          .WORD   200                     ; TEST FPS
14056 067654  000204                          .WORD   204                     ;RESULTANT FPS
14057                                 ;2/FSRC=0
14058 067656  005037  003030                  CLR     B#FLAG                  ;NO INTERRUPT
14059 067662  004767  000510                  JSR     R7,MLDSUB                       ;DO TEST
14060 067666  077777  000000  000000          .WORD   77777,0,0,0             ;AC0
      067674  000000
14061 067676  000000  000000  000000          .WORD   0,0,0,0 ;FSRC
      067704  000000
```

FLOATING POINT TESTS

```
14062 067706  000000  000000  000000          .WORD   0,0,0,0          ;RESULT
      067714  000000
14063 067716  007700                          .WORD   7700             ; TEST FPS
14064 067720  007704                          .WORD   7704             ;RESULTANT FPS
14065                               ;3/AC=1
14066 067722  005037  003030                   CLR     @#FLAG           ;NO INTERRUPT
14067 067726  004767  000444                   JSR     R7,MLDSUB                ;DO TEST
14068 067732  040200  000000  000000          .WORD   40200,0,0,0      ;AC0
      067740  000000
14069 067742  000277  177777  177777          .WORD   277,-1,-1,-1     ;FSRC
      067750  177777
14070 067752  000277  177777  177777          .WORD   277,-1,-1,-1             ;RESULT
      067760  177777
14071 067762  007717                          .WORD   7717             ; TEST FPS
14072 067764  007700                          .WORD   7700             ;RESULTANT FPS
14073                               ;4/AC>FSRC>0, TRUNCATE
14074 067766  005037  003030                   CLR     @#FLAG           ;NO INTERRUPT
14075 067772  004767  000400                   JSR     R7,MLDSUB                ;DO TEST
14076 067776  065500  000000  000000          .WORD   65500,0,0,1      ;AC0
      070004  000001
14077 070006  037577  177777  177777          .WORD   37577,-1,-1,-2   ;FSRC
      070014  177776
14078 070016  065077  177777  177777          .WORD   65077,-1,-1,-1           ;RESULT
      070024  177777
14079 070026  007717                          .WORD   7717             ; TEST FPS
14080 070030  007700                          .WORD   7700             ;RESULTANT FPS
14081                               ;5/AC<FSRC<0
14082 070032  005037  003030                   CLR     @#FLAG           ;NO INTERRUPT
14083 070036  004767  000334                   JSR     R7,MLDSUB                ;DO TEST
14084 070042  137577  177777  177777          .WORD   137577,-1,-1,-2 ;AC0
      070050  177776
14085 070052  165400  000000  000000          .WORD   165400,0,0,1     ;FSRC
      070060  000001
14086 070062  065000  000000  000000          .WORD   65000,0,0,0              ;RESULT
      070070  000000
14087 070072  007717                          .WORD   7717             ; TEST FPS
14088 070074  007700                          .WORD   7700             ;RESULTANT FPS
14089                               ;6/AC>FSRC>0
14090 070076  005037  003030                   CLR     @#FLAG           ;NO INTERRUPT
14091 070102  004767  000270                   JSR     R7,MLDSUB                ;DO TEST
14092 070106  017500  000000  000000          .WORD   17500,0,0,0      ;AC0
      070114  000000
14093 070116  123652  125252  125252          .WORD   123652,125252,125252,125252      ;FSRC
      070124  125252
14094 070126  103177  177777  177777          .WORD   103177,-1,-1,-1          ;RESULT
      070134  177777
14095 070136  000200                          .WORD   200              ; TEST FPS
14096 070140  000210                          .WORD   210              ;RESULTANT FPS
14097                               ;7/UNDERFLOW, TRAPS DISABLED
14098 070142  005037  003030                   CLR     @#FLAG                   ;NO INTERRUPT
14099 070146  004767  000224                   JSR     R7,MLDSUB                ;DO TEST
14100 070152  000300  000000  000000          .WORD   300,0,0,252              ;AC0
      070160  000252
14101 070162  000377  000001  000002          .WORD   377,1,2,3                ;FSRC
      070170  000003
14102 070172  000000  000000  000000          .WORD   0,0,0,0          ;RESULT
      070200  000000
```

FLOATING POINT TESTS

```
14103 070202  005740                        .WORD   5740            ; TEST FPS
14104 070204  005744                        .WORD   5744            ;RESULT FPS
14105                               ;8/UNDERFLOW, TRAP ENABLED
14106 070206  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
14107 070214  004767  000156                JSR     R7,MLDSUB               ;DO TEST
14108 070220  100277  000001  000002        .WORD   100277,1,2,-1           ;AC0
      070226  177777
14109 070230  100300  000001  000001        .WORD   100300,1,1,1            ;FSRC
      070236  000001
14110 070240  040417  040001  077403        .WORD   40417,40001,77403,0             ;RESULT
      070246  000000
14111 070250  002217                        .WORD   2217            ; TEST FPS
14112 070252  102200                        .WORD   102200          ;RESULT FPS
14113 070254  000012                        .WORD   12              ;FEC
14114                               ;9/OVERFLOW, TRAPS DISABLED
14115 070256  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
14116 070262  004767  000110                JSR     R7,MLDSUB               ;DO TEST
14117 070266  177777  177777  177777        .WORD   -1,-1,-1,-1             ;AC0
      070274  177777
14118 070276  040200  177777  177777        .WORD   40200,-1,-1,-1          ;FSRC
      070304  177777
14119 070306  000000  000000  000000        .WORD   0,0,0,0         ;RESULT
      070314  000000
14120 070316  006740                        .WORD   6740            ; TEST FPS
14121 070320  006746                        .WORD   6746            ;RESULT FPS
14122                               ;10/OVERFLOW, TRAPS ENABLED
14123 070322  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
14124 070330  004767  000042                JSR     R7,MLDSUB               ;DO TEST
14125 070334  157700  025252  025252        .WORD   157700,25252,25252,25252                ;AC0
      070342  025252
14126 070344  167700  000000  000000        .WORD   167700,0,0,0            ;FSRC
      070352  000000
14127 070354  007420  017777  117777        .WORD   7420,017777,117777,117777               ;RESULT
      070362  117777
14128 070364  001240                        .WORD   1240            ; TEST FPS
14129 070366  101242                        .WORD   101242          ;RESULT FPS
14130 070370  000010                        .WORD   10              ;FEC
14131                               ;
14132
14133 070372  000167  000212                JMP     HOP13
14134
14135                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14136                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14137                               ;
14138                               ;               AC0
14139                               ;               FSRC
14140                               ;               FPS BEFORE EXECUTION
14141                               ;               FPS AFTER EXECUTION
14142                               ;               (FEC)
14143                               ;
14144                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14145                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14146                               ;
14147 070376  012605                MLDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14148 070400  012737  070454  000244        MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14149 070406  012702  000200                MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14150 070412  170102                        LDFPS   R2                      ;LOAD FPS
```

FLOATING POINT TESTS

```
14151 070414  172415                        LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14152 070416  010501                        MOV     R5,R1                   ;POINT TO FSRC DATA
14153 070420  062701  000010                ADD     #10,R1
14154 070424  016502  000030                MOV     30(R5),R2               ;GET TEST FPS
14155 070430  170102                        LDFPS   R2                      ;LOAD TEST FPS
14156                               ;
14157 070432  171011                        MULD    (R1),AC0                ;*TEST INSTRUCTION
14158 070434  170011                1$:     SETD                            ;WAIT FOR POSSIBLE FPA TRAP.
14159                               ;
14160                               ;INSTRUCTION DIDNT TRAP
14161 070436  032737  000001  003030        BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
14162 070444  001420                        BEQ     2$                      ;BRANCH IF GOOD
14163 070446  104003                        ERROR   .3                  ;FPP ERROR
14164                                                                   ;INSTRUCTION SHOULD HAVE TRAPPED
14165 070450  000167  000032                JMP     2$                      ;REJOIN CODE
14166                               ;
14167                               ;INSTRUCTION TRAPPED
14168 070454  032737  000001  003030  50$:   BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14169 070462  001003                        BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14170 070464  104003                        ERROR   .3                  ;FPP ERROR
14171                                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
14172 070466  000167  000014                JMP     2$                      ;REJOIN CODE
14173 070472  012604                51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14174 070474  005726                        TST     (SP)+                   ;CLEAN UP STACK
14175 070476  022704  070434                CMP     #1$,R4                  ;
14176 070502  001401                        BEQ     2$                      ;BRANCH IF GOOD COMPARE
14177 070504  104003                        ERROR   .3                  ;FPP ERROR
14178                                                                   ;PC WAS INCORRECT
14179                               ;
14180                               ;COMMON CODE FOR TRAP AND NO TRAP
14181 070506  170203                2$:     STFPS   R3                      ;SAVE FPS
14182 070510  012702  000200                MOV     #200,R2                 ;SET FPP TO DOUBLE
14183 070514  170102                        LDFPS   R2
14184 070516  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14185 070522  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
14186 070524  026503  000032                CMP     32(R5),R3               ;VERIFY STATUS
14187 070530  001401                        BEQ     3$                      ;BRANCH IF GOOD
14188 070532  104003                        ERROR   .3                  ;FPP ERROR
14189                                                                   ;BAD FPS
14190 070534  010504                3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14191 070536  062704  000020                ADD     #20,R4
14192 070542  004767  044646          4$:   JSR     R7,DATVER                       ;VERIFY DATA
14193 070546  005767  112346                TST     COUNT
14194 070552  001401                        BEQ     5$                      ;BRANCH IF GOOD
14195 070554  104003                        ERROR   .3                  ;FPP ERROR
14196                                                                   ;BAD AC0
14197 070556  005737  003030          5$:   TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14198 070562  001002                        BNE     6$                      ;BRANCH IF NEED TO CHECK
14199 070564  000165  000034                JMP     34(R5)                  ;RETURN FROM TEST
14200                               ;VERIFY ERROR STATUS
14201 070570  170301                6$:     STST    R1                      ;SAVE FEC
14202 070572  016504  000034                MOV     34(R5),R4               ;GET FEC
14203 070576  020401                        CMP     R4,R1                   ;VERIFY FEC
14204 070600  001401                        BEQ     7$                      ;BRANCH IF GOOD
14205 070602  104003                        ERROR   .3                  ;FPP ERROR
14206                                                                   ;BAD FEC
14207 070604  000165  000036          7$:   JMP     36(R5)                  ;RETURN FROM TEST
```

FLOATING POINT TESTS

```
14208 070610                          MOP13:
14211                                 ;
14212                                 ;
14213                                 ;---------------------------------
14214                                 ;TEST MODF
14215                                 ;
14216                                         ;
14217                                         ;
14218                                         ;
14219 070610                          MMODF:
14220
14221                                 ;
14222                                 ;1/AC=0 FSRC=0
14223 070610  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14224 070614  004767  000554                  JSR     R7,MDFSUB               ;DO TEST
14225 070620  000100  000000                  .WORD   100,0   ;AC0
14226 070624  012346  177777                  .WORD   12346,-1        ;FSRC
14227 070630  000000  000000                  .WORD   0,0             ;FRACTIONAL RESULT
14228 070634  000000  000000                  .WORD   0,0             ;INTEGER RESULT
14229 070640  000013                          .WORD   13              ; TEST FPS
14230 070642  000004                          .WORD   4               ;RESULTANT FPS
14231                                 ;2/FSRC=0
14232 070644  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14233 070650  004767  000520                  JSR     R7,MDFSUB               ;DO TEST
14234 070654  012356  177777                  .WORD   12356,-1        ;AC0
14235 070660  000000  000000                  .WORD   0,0     ;FSRC
14236 070664  000000  000000                  .WORD   0,0             ;FRACTIONAL RESULT
14237 070670  000000  000000                  .WORD   0,0             ;INTEGER RESULT
14238 070674  000003                          .WORD   3               ; TEST FPS
14239 070676  000004                          .WORD   4       ;RESULTANT FPS
14240                                 ;3/AC=0
14241 070700  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14242 070704  004767  000464                  JSR     R7,MDFSUB               ;DO TEST
14243 070710  000000  000000                  .WORD   0,0             ;AC0
14244 070714  177777  177777                  .WORD   -1,-1           ;FSRC
14245 070720  000000  000000                  .WORD   0,0             ;FRACTIONAL  RESULT
14246 070724  000000  000000                  .WORD   0,0             ;INTEGER RESULT
14247 070730  007500                          .WORD   7500            ; TEST FPS
14248 070732  007504                          .WORD   7504            ;RESULT FPS
14249                                 ;4/AC>FSRC>0
14250 070734  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14251 070740  004767  000430                  JSR     R7,MDFSUB               ;DO TEST
14252 070744  046252  125252                  .WORD   46252,125252    ;AC0
14253 070750  040300  000000                  .WORD   40300,0 ;FSRC
14254 070754  000000  000000                  .WORD   0,0             ;FRACTIONAL RESULT
14255 070760  046377  177777                  .WORD   46377,-1                ;INTEGER RESULT
14256 070764  000013                          .WORD   13              ; TEST FPS
14257 070766  000004                          .WORD   4               ;RESULTANT FPS
14258                                 ;5/AC>FSRC>0
14259 070770  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14260 070774  004767  000374                  JSR     R7,MDFSUB               ;DO TEST
14261 071000  077652  125252                  .WORD   77652,125252    ;AC0
14262 071004  040300  000000                  .WORD   40300,0 ;FSRC
14263 071010  000000  000000                  .WORD   0,0             ;FRACTIONAL RESULT
14264 071014  077777  177777                  .WORD   77777,-1                ;INTEGER RESULT
14265 071020  000000                          .WORD   0               ; TEST FPS
14266 071022  000004                          .WORD   4               ;RESULTANT FPS
```

FLOATING POINT TESTS

```
14267
14268
14269 071024  005037  003030                ;6/AC>0<FSRC, INTEGERS
                            CLR     @#FLAG          ;NO INTERRUPT
14270 071030  004767  000340                JSR     R7,MDFSUB
                                                    ;DO TEST
14271 071034  060600  000000                .WORD   60600,0         ;AC0
14272 071040  147400  025700                .WORD   147400,25700
                                                    ;FSRC
14273 071044  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14274 071050  170000  025700                .WORD   170000,25700
                                                    ;INTEGER RESULT
14275 071054  007400                        .WORD   7400            ; TEST FPS
14276 071056  007404                        .WORD   7404            ;RESULT FPS
14277
14278 071060  005037  003030                ;7/AC<0<FSRC, FRACTIONAL
                            CLR     @#FLAG          ;NO INTERRUPT
14279 071064  004767  000304                JSR     R7,MDFSUB
                                                    ;DO TEST
14280 071070  100227  177777                .WORD   100227,-1
                                                    ;AC0
14281 071074  044025  025252                .WORD   44025,25252
                                                    ;FSRC
14282 071100  104061  021251                .WORD   104061,21251
                                                    ;FRACTIONAL  RESULT
14283 071104  000000  000000                .WORD   0,0             ;INTEGER RESULT
14284 071110  000000                        .WORD   0               ; TEST FPS
14285 071112  000010                        .WORD   10              ;RESULT FPS
14286
14287 071114  005037  003030                ;8/AC<0>FSRC, TRUNCATE
                            CLR     @#FLAG          ;NO INTERRUPT
14288 071120  004767  000250                JSR     R7,MDFSUB
                                                    ;DO TEST
14289 071124  046252  125252                .WORD   46252,125252
                                                    ;AC0
14290 071130  040300  000000                .WORD   40300,0         ;FSRC
14291 071134  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14292 071140  046377  177777                .WORD   46377,-1
                                                    ;INTEGER RESULT
14293 071144  000053                        .WORD   53              ; TEST FPS
14294 071146  000044                        .WORD   44              ;RESULT FPS
14295
14296 071150  005037  003030                ;9/ROUND INTEGER
                            CLR     @#FLAG          ;NO INTERRUPT
14297 071154  004767  000214                JSR     R7,MDFSUB
                                                    ;DO TEST
14298 071160  046252  125252                .WORD   46252,125252
                                                    ;AC0
14299 071164  040300  000000                .WORD   40300,0         ;FSRC
14300 071170  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14301 071174  046377  177777                .WORD   46377,-1
                                                    ;INTEGER RESULT
14302 071200  000013                        .WORD   13              ; TEST FPS
14303 071202  000004                        .WORD   4               ;RESULT FPS
14304
14305 071204  005037  003030                ;10/TRUNCATE FRACTION
                            CLR     @#FLAG          ;NO INTERRUPT
14306 071210  004767  000160                JSR     R7,MDFSUB
                                                    ;DO TEST
14307 071214  040777  177777                .WORD   40777,-1
                                                    ;AC0
14308 071220  040200  000000                .WORD   40200,0         ;FSRC
14309 071224  040177  177770                .WORD   40177,177770
                                                    ;FRACTIONAL  RESULT
14310 071230  040740  000000                .WORD   40740,0         ;INTEGER RESULT
14311 071234  000000                        .WORD   0               ; TEST FPS
14312 071236  000000                        .WORD   0               ;RESULT FPS
14313
14314 071240  005037  003030                ;11/ROUND INTEGER
                            CLR     @#FLAG          ;NO INTERRUPT
14315 071244  004767  000124                JSR     R7,MDFSUB
                                                    ;DO TEST
14316 071250  000000  000000                .WORD   0,0             ;AC0
14317 071254  000000  000000                .WORD   0,0             ;FSRC
14318 071260  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14319 071264  000000  000000                .WORD   0,0             ;INTEGER RESULT
14320 071270  000000                        .WORD   0               ; TEST FPS
14321 071272  000004                        .WORD   4               ;RESULT FPS
14322
14323 071274  005037  003030                ;12/ROUND FRACTION
                            CLR     @#FLAG          ;NO INTERRUPT
```

FLOATING POINT TESTS

```
14324 071300  004767  000070                  JSR     R7,MDFSUB               ;DO TEST
14325 071304  040225  125252                  .WORD   40225,125252            ;AC0
14326 071310  066652  052525                  .WORD   66652,52525             ;FSRC
14327 071314  000000  000000                  .WORD   0,0                 ;FRACTIONAL  RESULT
14328 071320  066707  025160                  .WORD   66707,25160             ;INTEGER RESULT
14329 071324  007027                          .WORD   7027                ; TEST FPS
14330 071326  007004                          .WORD   7004                ;RESULT FPS
14331                               ;/OVERFLOW
14332 071330  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
14333 071336  004767  000032                  JSR     R7,MDFSUB               ;DO TEST
14334 071342  076000  000000                  .WORD   76000,0             ;AC0
14335 071346  076000  000000                  .WORD   76000,0             ;FSRC
14336 071352  000000  000000                  .WORD   0,0                 ;FRACTIONAL  RESULT
14337 071356  033600  000000                  .WORD   33600,0             ;INTEGER RESULT
14338 071362  001000                          .WORD   1000                ; TEST FPS
14339 071364  101006                          .WORD   101006              ;RESULT FPS
14340 071366  000010                          .WORD   10                  ;FEC
14341                               ;
14342 071370  000167  000254                  JMP     HOP14
14343
14344                               ;*X*X*X
```

FLOATING POINT TESTS

```
14381 071460  032737  000001  003030  50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14382 071466  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14383 071470  104003                          ERROR   .+3             ;FPP ERROR
14384                                         ;INSTRUCTION WASNT SUPPOSE TO TRAP
14385 071472  000167  000014                  JMP     2$                      ;REJOIN CODE
14386 071476  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14387 071500  005726                          TST     (SP)+                   ;CLEAN UP STACK
14388 071502  022704  071440                  CMP     #1$,R4                  ;
14389 071506  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
14390 071510  104003                          ERROR   .+3             ;FPP ERROR
14391                                         ;PC WAS INCORRECT
14392                                 ;
14393                                 ;COMMON CODE FOR TRAP AND NO TRAP
14394 071512  170203                  2$:     STFPS   R3                      ;SAVE FPS
14395 071514  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
14396 071520  170102                          LDFPS   R2
14397 071522  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14398                                 ;SAVE FRACTIONAL RESULT
14399 071526  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
14400 071530  026503  000022                  CMP     22(R5),R3               ;VERIFY STATUS
14401 071534  001401                          BEQ     3$                      ;BRANCH IF GOOD
14402 071536  104003                          ERROR   .+3             ;FPP ERROR
14403                                         ;BAD FPS
14404 071540  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14405 071542  062704  000010                  ADD     #10,R4
14406 071546  004767  043624          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
14407 071552  005767  111342                  TST     COUNT
14408 071556  001401                          BEQ     5$                      ;BRANCH IF GOOD
14409 071560  104003                          ERROR   .+3             ;FPP ERROR
14410                                         ;BAD AC0
14411                                 ;SAVE INTEGER RESULT
14412 071562  174111                  5$:     STD     AC1,(R1)                ;SAVE AC1 RESULT
14413 071564  010504                          MOV     R5,R4           ;POINT TO EXPECTED
14414 071566  062704  000014                  ADD     #14,R4
14415 071572  004767  043600                  JSR     R7,DATVFR               ;VERIFY DATA
14416 071576  005767  111316                  TST     COUNT                   ;
14417 071602  001401                          BEQ     6$                      ;BRANCH IF GOOD
14418 071604  104003                          ERROR   .+3             ;FPP ERROR
14419                                         ;BAD AC1
14420 071606  005737  003030          6$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14421 071612  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
14422 071614  000165  000024                  JMP     24(R5)                  ;RETURN FROM TEST
14423 071620  170301                  7$:     STST    R1                      ;SAVE FEC
14424 071622  016504  000024                  MOV     24(R5),R4               ;GET FEC
14425 071626  020401                          CMP     R4,R1                   ;VERIFY FEC
14426 071630  001401                          BEQ     8$                      ;BRANCH IF GOOD
14427 071632  104003                          ERROR   .+3             ;FPP ERROR
14428                                         ;BAD FEC
14429 071634  000165  000026          8$:     JMP     26(R5)          ;RETURN FROM TEST
14430                                 ;
14431 071640  177777  177777  177777  MODGAR: .WORD   -1,-1,-1,-1             ;KNOWN DATA FOR AC1
      071646  177777
14432 071650                          MOP14:
14433
14436                                 ;
14437                                 ;
14438                                 ;---------------------------------
```

FLOATING POINT TESTS

```
14439                                   ;TEST MODD
14440                                   ;
14441                                           ;
14442                                           ;
14443                                           ;
14444 071650                            MMODD:
14445
14446                                   ;
14447                                   ;1/AC>FSRC=0
14448 071650  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14449 071654  004767  001164                      JSR     R7,MDDSUB               ;DO TEST
14450 071660  012345  177777  177777              .WORD   12345,-1,-1,-1  ;AC0
      071666  177777
14451 071670  000100  000000  000000              .WORD   100,0,0,0       ;FSRC
      071676  000000
14452 071700  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      071706  000000
14453 071710  000000  000000  000000              .WORD   0,0,0,0         ;INTEGER RESULT
      071716  000000
14454 071720  000200                            .WORD   200             ; TEST FPS
14455 071722  000204                            .WORD   204             ;RESULTANT FPS
14456                                   ;2/AC=0<FSRC
14457 071724  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14458 071730  004767  001110                      JSR     R7,MDDSUB               ;DO TEST
14459 071734  000000  000000  000000              .WORD   0,0,0,0 ;AC0
      071742  000000
14460 071744  001234  177777  000000              .WORD   1234,-1,0,0     ;FSRC
      071752  000000
14461 071754  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      071762  000000
14462 071764  000000  000000  000000              .WORD   0,0,0,0         ;INTEGER RESULT
      071772  000000
14463 071774  007717                            .WORD   7717            ; TEST FPS
14464 071776  007704                            .WORD   7704            ;RESULTANT FPS
14465                                   ;3/AC>FSRC>0
14466 072000  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14467 072004  004767  001034                      JSR     R7,MDDSUB               ;DO TEST
14468 072010  056252  125252  125252              .WORD   56252,125252,125252,125250      ;AC0
      072016  125250
14469 072020  040300  000000  000000              .WORD   40300,0,0,0     ;FSRC
      072026  000000
14470 072030  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      072036  000000
14471 072040  056377  177777  177777              .WORD   56377,-1,-1,-4          ;INTEGER RESULT
      072046  177774
14472 072050  000213                            .WORD   213             ; TEST FPS
14473 072052  000204                            .WORD   204             ;RESULTANT FPS
14474                                   ;4/AC<0>FSRC
14475 072054  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14476 072060  004767  000760                      JSR     R7,MDDSUB               ;DO TEST
14477 072064  140240  000000  000000              .WORD   140240,0,0,0    ;AC0
      072072  000000
14478 072074  063714  146314  133572              .WORD   63714,146314,133572,167737      ;FSRC
      072102  167737
14479 072104  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      072112  000000
14480 072114  163777  177777  162531              .WORD   163777,-1,162531,125726         ;INTEGER RESULT
```

FLOATING POINT TESTS

```
        072122  125726
14481   072124  000210                          .WORD   210             ; TEST FPS
14482   072126  000204                          .WORD   204             ;RESULTANT FPS
14483                           ;5/AC>FSRC>0
14484   072130  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14485   072134  004767  000704                  JSR     R7,MDDSUB               ;DO TEST
14486   072140  056200  000000  000000          .WORD   56200,0,0,1     ;AC0
        072146  000001
14487   072150  040340  000000  000000          .WORD   40340,0,0,0     ;FSRC
        072156  000000
14488   072160  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL RESULT
        072166  000000
14489   072170  056340  000000  000000          .WORD   56340,0,0,1             ;INTEGER RESULT
        072176  000001
14490   072200  000213                          .WORD   213             ; TEST FPS
14491   072202  000204                          .WORD   204             ;RESULTANT FPS
14492                           ;6/TRUNCATE
14493   072204  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14494   072210  004767  000630                  JSR     R7,MDDSUB               ;DO TEST
14495   072214  056252  125252  125252          .WORD   56252,125252,125252,125252              ;AC0
        072222  125252
14496   072224  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
        072232  000000
14497   072234  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
        072242  000000
14498   072244  056377  177777  177777          .WORD   56377,-1,-1,-1          ;INTEGER RESULT
        072252  177777
14499   072254  000253                          .WORD   253             ; TEST FPS
14500   072256  000244                          .WORD   244             ;RESULT FPS
14501                           ;7/TRUNCATE FRACTION
14502   072260  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14503   072264  004767  000554                  JSR     R7,MDDSUB               ;DO TEST
14504   072270  023252  125252  125252          .WORD   23252,125252,125252,125252              ;AC0
        072276  125252
14505   072300  040300  000000  000000          .WORD   40300,0,0,0             ;FSRC
        072306  000000
14506   072310  023377  177777  177777          .WORD   23377,-1,-1,-1          ;FRACTIONAL  RESULT
        072316  177777
14507   072320  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
        072326  000000
14508   072330  000253                          .WORD   253             ; TEST FPS
14509   072332  000240                          .WORD   240             ;RESULT FPS
14510                           ;8/ROUND INTEGER
14511   072334  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
14512   072340  004767  000500                  JSR     R7,MDDSUB               ;DO TEST
14513   072344  076600  000000  000000          .WORD   76600,0,0,125252                ;AC0
        072352  125252
14514   072354  040300  000000  000000          .WGRD   40300,0,0,0             ;FSRC
        072362  000000
14515   072364  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
        072372  000000
14516   072374  076700  000000  000000          .WORD   76700,0,0,-1            ;INTEGER RESULT
        072402  177777
14517   072404  000200                          .WORD   200             ; TEST FPS
14518   072406  000204                          .WORD   204             ;RESULT FPS
14519                           ;9/ROUND THROUGH FRACTION
14520   072410  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
```

FLOATING POINT TESTS

```
14521 072414  004767  000424                JSR     R7,MDDSUB                       ;DO TEST
14522 072420  041525  052525  052525        .WORD   41525,052525,52525,52525                ;ACO
      072426  052525
14523 072430  040300  000000  000000        .WORD   40300,0,0,0                     ;FSRC
      072436  000000
14524 072440  040177  177777  177777        .WORD   40177,-1,-1,177740                      ;FRACTIONAL  RESULT
      072446  177740
14525 072450  041636  000000  000000        .WORD   41636,0,0,0                     ;INTEGER RESULT
      072456  000000
14526 072460  007700                        .WORD   7700                    ; TEST FPS
14527 072462  007700                        .WORD   7700                    ;RESULT FPS
14528                       ;/OVERFLOW, TRAPS ENABLED
14529 072464  012737  000001  003030        MOV     #1,@#FLAG                       ;INTERRUPT
14530 072472  004767  000346                JSR     R7,MDDSUB                       ;DO TEST
14531 072476  177777  177777  177777        .WORD   -1,-1,-1,-1                     ;ACO
      072504  177777
14532 072506  040400  000000  000000        .WORD   40400,0,0,0                     ;FSRC
      072514  000000
14533 072516  000000  000000  000000        .WORD   0,0,0,0                 ;FRACTIONAL  RESULT
      072524  000000
14534 072526  100177  177777  177777        .WORD   100177,-1,-1,-1                 ;INTEGER RESULT
      072534  177777
14535 072536  007700                        .WORD   7700                    ; TEST FPS
14536 072540  107706                        .WORD   107706                  ;RESULT FPS
14537 072542  000010                        .WORD   10                      ;FEC
14538                       ;/INTEGER CHOPPED TO 56 BITS
14539 072544  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
14540 072550  004767  000270                JSR     R7,MDDSUB                       ;DO TEST
14541 072554  056700  000000  000000        .WORD   56700,0,0,-1                    ;ACO
      072562  177777
14542 072564  044440  177777  177777        .WORD   44440,-1,-1,-1                  ;FSRC
      072572  177777
14543 072574  000000  000000  000000        .WORD   0,0,0,0                 ;FRACTIONAL  RESULT
      072602  000000
14544 072604  063161  100000  000001        .WORD   63161,100000,1,40775                    ;INTEGER RESULT
      072612  040775
14545 072614  000200                        .WORD   200                     ; TEST FPS
14546 072616  000204                        .WORD   204                     ;RESULT FPS
14547                       ;/OVERFLOW, TRAPS DISABLED
14548 072620  012737  000002  003030        MOV     #2,@#FLAG                       ;NO INTERRUPT
14549 072626  004767  000212                JSR     R7,MDDSUB                       ;DO TEST
14550 072632  066600  000000  000000        .WORD   66600,0,0,0                     ;ACO
      072640  000000
14551 072642  066600  000000  000000        .WORD   66600,0,0,0                     ;FSRC
      072650  000000
14552 072652  000000  000000  000000        .WORD   0,0,0,0                 ;FRACTIONAL  RESULT
      072660  000000
14553 072662  015200  000000  000000        .WORD   15200,0,0,0                     ;INTEGER RESULT
      072670  000000
14554 072672  047700                        .WORD   47700                   ; TEST FPS
14555 072674  147706                        .WORD   147706                  ;RESULT FPS
14556 072676  000010                        .WORD   10                      ;FEC
14557                       ;/UNDERFLOW, TRAPS DISABLED
14558 072700  012737  000002  003030        MOV     #2,@#FLAG                       ;NO INTERRUPT
14559 072706  004767  000132                JSR     R7,MDDSUB                       ;DO TEST
14560 072712  100277  000001  000002        .WORD   100277,1,2,-1                   ;ACO
      072720  177777
```

```
072722  100300  000001  000001          .WORD   100300,1,1,1            ;FSRC
072730  000001
072732  000000  000000  000000          .WORD   0,0,0,0         ;FRACTIONAL  RESULT
072740  000000
072742  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
072750  000000
072752  005200                          .WORD   5200            ; TEST FPS
072754  005204                          .WORD   5204            ;RESULT FPS
072756  000010                          .WORD   10              ;FEC
                                ;/UNDERFLOW TRAPS ENABLED, UV AS RESULT
072760  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
072766  004767  000052                  JSR     R7,MDDSUB               ;DO TEST
072772  100277  000001  000002          .WORD   100277,1,2,-1           ;AC0
073000  177777
073002  100300  000001  000001          .WORD   100300,1,1,1            ;FSRC
073010  000001
073012  040417  040001  077403          .WORD   40417,40001,77403,0             ;FRACTIONAL  RESULT
073020  000000
073022  000000  000000  000000          .WORD   0,0,0,0         ;INTEGER RESULT
073030  000000
073032  002200                          .WORD   2200            ; TEST FPS
073034  102200                          .WORD   102200          ;RESULT FPS
073036  000012                          .WORD   12              ;FEC
                                ;

073040  000167  000244                  JMP     HOP15           ;JUMP 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
                                ;
                                ;               AC0
                                ;               FSRC
                                ;               FRACTIONAL RESULT
                                ;               INTEGER RESULT
                                ;               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
                                ;
073044  012605                  MDDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
073046  012737  073130  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
073054  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
073060  170102                          LDFPS   R2                      ;LOAD FPS
073062  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
073064  012701  071640                  MOV     #MODGAR,R1              ;LOAD KNOWN INTO AC1
073070  172511                          LDD     (R1),AC1                ;
073072  010501                          MOV     R5,R1                   ;POINT TO FSRC DATA
073074  062701  000010                  ADD     #10,R1
073100  016502  000040                  MOV     40(R5),R2               ;GET TEST FPS
073104  170102                          LDFPS   R2                      ;LOAD TEST FPS
                                ;
073106  171411                          MODD    (R1),AC0                ;*TEST INSTRUCTION
073110  170011                  1$:     SETD                            ;WAIT FOR POSSIBLE FPA TRAP.
                                ;
                                ;INSTRUCTION DIDNT TRAP
073112  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
```

FLOATING POINT TESTS

```
14611 073120  001420                        BEQ     2$                  ;BRANCH IF GOOD
14612 073122  104003                        ERROR   .3                ;FPP ERROR
14613                                                                 ;INSTRUCTION SHOULD HAVE TRAPPED
14614 073124  000167  000032                JMP     2$                          ;REJOIN CODE
14615                               ;
14616                               ;INSTRUCTION TRAPPED
14617 073130  032737  000001  003030  50$:   BIT     #1,@#FLAG                   ;SEE IF EXPECTING A TRAP
14618 073136  001003                        BNE     51$                         ;BRANCH IF EXPECTING A TRAP
14619 073140  104003                        ERROR   .3                ;FPP ERROR
14620                                                                 ;INSTRUCTION WASNT SUPPOSE TO TRAP
14621 073142  000167  000014                JMP     2$                          ;REJOIN CODE
14622 073146  012604                51$:    MOV     (SP)+,R4                    ;SEE IF PC = INSTRUCTION
14623 073150  005726                        TST     (SP)+                       ;CLEAN UP STACK
14624 073152  022704  073110                CMP     #1$,R4                      ;
14625 073156  001401                        BEQ     2$                          ;BRANCH IF GOOD COMPARE
14626 073160  104003                        ERROR   .3                ;FPP ERROR
14627                                                                 ;PC WAS INCORRECT
14628                               ;
14629                               ;COMMON CODE FOR TRAP AND NO TRAP
14630 073162  170203                2$:     STFPS   R3                          ;SAVE FPS
14631 073164  012702  000200                MOV     #200,R2                     ;SET FPP TO DOUBLE
14632 073170  170102                        LDFPS   R2
14633 073172  012701  003142                MOV     #RECDST,R1                  ;POINT TO RECEIVED DATA TABLE
14634                               ;SAVE FRACTIONAL RESULT
14635 073176  174011                        STD     AC0,(R1)                    ;SAVE AC0  RESULT
14636 073200  026503  000042                CMP     42(R5),R3                   ;VERIFY STATUS
14637 073204  001401                        BEQ     3$                          ;BRANCH IF GOOD
14638 073206  104003                        ERROR   .3                ;FPP ERROR
14639                                                                 ;BAD FPS
14640 073210  010504                3$:     MOV     R5,R4
                                                                            ;POINT TO EXPECTED DATA
14641 073212  062704  000020                ADD     #20,R4
14642 073216  004767  042172        4$:     JSR     R7,DATVER                           ;VERIFY DATA
14643 073222  005767  107672                TST     COUNT
14644 073226  001401                        BEQ     5$                          ;BRANCH IF GOOD
14645 073230  104003                        ERROR   .3                ;FPP ERROR
14646                                                                 ;BAD AC0
14647                               ;SAVE INTEGER RESULT
14648 073232  174111                5$:     STD     AC1,(R1)                    ;SAVE AC1 RESULT
14649 073234  010504                        MOV     R5,R4             ;POINT TO EXPECTED
14650 073236  062704  000030                ADD     #30,R4
14651 073242  004767  042146                JSR     R7,DATVER                   ;VERIFY DATA
14652 073246  005767  107646                TST     COUNT                       ;
14653 073252  001401                        BEQ     6$                          ;BRANCH IF GOOD
14654 073254  104003                        ERROR   .3                ;FPP ERROR
14655                                                                 ;BAD AC1
14656 073256  005737  003030        6$:     TST     @#FLAG                      ;SEE IF NEED TO CHECK FEC
14657 073262  001002                        BNE     7$                          ;BRANCH IF NEED TO CHECK
14658 073264  000165  000044                JMP     44(R5)                      ;RETURN FROM TEST
14659 073270  170301                7$:     STST    R1                          ;SAVE FEC
14660 073272  016504  000044                MOV     44(R5),R4                   ;GET FEC
14661 073276  020401                        CMP     R4,R1                       ;VERIFY FEC
14662 073300  001401                        BEQ     8$                          ;BRANCH IF GOOD
14663 073302  104003                        ERROR   .3                ;FPP ERROR
14664                                                                           ;BAD FEC
14665 073304  000165  000046        8$:     JMP     46(R5)            ;RETURN FROM TEST
14666                               ;
14667 073310                        HOP15:
```

```
                                        ;
                                        ;
                                        ;--------------------------------
                                        ;TEST STCFD
                                        ;
                                                ;
                                                ;
                                                ;
073310                                  MSFD:

                                        ;
                                        ;1/AC=0
073310  004767  000170                          JSR     R7,SFDSUB                       ;DO TEST
073314  000177  000000  000000                  .WORD   0177,0,0,1                      ;AC0
        073322  000001
073324  000000  000000  000000                  .WORD   0,0,0,0                 ;RESULT
        073332  000000
073334  047557                                  .WORD   47557                   ; TEST FPS
073336  047544                                  .WORD   47544                   ;RESULT FPS
                                        ;2/AC>0, TRUNCATE
073340  004767  000140                          JSR     R7,SFDSUB               ;DO TEST
073344  077577  177777  177777                  .WORD   77577,-1,-1,-1                  ;AC0
        073352  177777
073354  077577  177777  000000                  .WORD   77577,-1,0,0                    ;RESULT
        073362  000000
073364  007540                                  .WORD   7540                    ; TEST FPS
073366  007540                                  .WORD   7540                    ;RESULT FPS
                                        ;3/AC<0, ROUND
073370  004767  000110                          JSR     R7,SFDSUB                       ;DO TEST
073374  100377  177777  100000                  .WORD   100377,-1,100000,0                      ;AC0
        073402  000000
073404  100377  177777  000000                  .WORD   100377,-1,0,0           ;RESULT
        073412  000000
073414  007517                                  .WORD   7517                    ; TEST FPS
073416  007510                                  .WORD   7510                    ;RESULT FPS
                                        ;4/AC=-0
073420  004767  000060                          JSR     R7,SFDSUB                       ;DO TEST
073424  100000  000000  000000                  .WORD   100000,0,0,0                    ;AC0
        073432  000000
073434  000000  000000  000000                  .WORD   0,0,0,0                 ;RESULT
        073442  000000
073444  007757                                  .WORD   7757                    ; TEST FPS
073446  007744                                  .WORD   7744                    ;RESULT FPS
                                        ;5/AC<0
073450  004767  000030                          JSR     R7,SFDSUB                       ;DO TEST
073454  125252  125252  125252                  .WORD   125252,125252,125252,125252                     ;AC0
        073462  125252
073464  125252  125252  000000                  .WORD   125252,125252,0,0                       ;RESULT
        073472  000000
073474  000000                                  .WORD   0                       ; TEST FPS
073476  000010                                  .WORD   10                      ;RESULT FPS
                                        ;


073500  000167  000104                          JMP     HOP16                           ;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
                                        ;STCFD
```

FLOATING POINT TESTS

```
14717                                   ;               AC0
14718                                   ;               RESULT
14719                                   ;               FPS BEFORE EXECUTION
14720                                   ;               FPS AFTER EXECUTION
14721                                   ;
14722                                   ;THERE CAN BE NO TRAPS WITH THE STCFD INSTRUCTION
14723                                   ;*X*X*X
```

```
073620  000177  000000  000000          .WORD   177,0,0,0                ;AC0
073626  000000
073630  000000  000000                  .WORD   0,0             ;RESULT
073634  000200                          .WORD   200             ; TEST FPS
073636  000204                          .WORD   204             ;RESULT FPS
                                ;2/AC=-0
073640  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
073644  004767  000170                  JSR     R7,SDFSUB                ;DO TEST
073650  100000  000300  000200          .WORD   100000,300,200,100              ;AC0
073656  000100
073660  000000  000000                  .WORD   0,0             ;RESULT
073664  007777                          .WORD   7777            ; TEST FPS
073666  007744                          .WORD   7744            ;RESULT FPS
                                ;3/AC>0, TRUNCATE
073670  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
073674  004767  000140                  JSR     R7,SDFSUB                ;DO TEST
073700  055555  055555  177777          .WORD   55555,55555,-1,-1               ;AC0
073706  177777
073710  055555  055555                  .WORD   55555,55555              ;RESULT
073714  000240                          .WORD   240             ; TEST FPS
073716  000240                          .WORD   240             ;RESULT FPS
                                ;4/AC<0, ROUND TO UNDEFINED VARIABLE
073720  012737  000001  003030          MOV     #1,@#FLAG                ;INTERRUPT
073726  004767  000106                  JSR     R7,SDFSUB                ;DO TEST
073732  077777  177777  100000          .WORD   77777,-1,100000,0               ;AC0
073740  000000
073742  000000  000000                  .WORD   0,0             ;RESULT
073746  001200                          .WORD   1200            ; TEST FPS
073750  101206                          .WORD   101206          ;RESULT FPS
                                ;5/AC<0, ROUND
073752  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
073756  004767  000056                  JSR     R7,SDFSUB                ;DO TEST
073762  125252  125252  125252          .WORD   125252,125252,125252,125252             ;AC0
073770  125252
073772  125252  125253                  .WORD   125252,125253            ;RESULT
073776  007700                          .WORD   7700            ; TEST FPS
074000  007710                          .WORD   7710            ;RESULT FPS
                                ;6/ROUND TO UV, TRAPS DISABLED
074002  012737  000002  003030          MOV     #2,@#FLAG                ;INTERRUPT
074010  004767  000024                  JSR     R7,SDFSUB                ;DO TEST
074014  077777  177777  177777          .WORD   77777,-1,-1,0            ;AC0
074022  000000
074024  000000  000000                  .WORD   0,0             ;RESULT
074030  006700                          .WORD   6700            ; TEST FPS
074032  006706                          .WORD   6706            ;RESULT FPS
                                ;


074034  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
                                ;STCDF
                                ;               AC0
                                ;               RESULT
                                ;               FPS BEFORE EXECUTION
                                ;               FPS AFTER EXECUTION
```

```
14827                                   ;
14828                                   ;A TRAP CAN ONLY OCCUR IF ROUNDING CAUSES OVERFLOW
14829                                   ;*X*X*X
```

FLOATING POINT TESTS

```
14884 074220  012704  003122          7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
14885 074224  170314                          STST    (R4)                    ;SAVE FEC
14886 074226  021427  000010                  CMP     (R4),#10                        ;VERIFY FEC FOR OVERFLOW
14887 074232  001401                          BEQ     8$                      ;BRANCH IF GOOD
14888 074234  104003                          ERROR   +3              ;FPP ERROR
14889                                                                 ;BAD FEC
14890 074236  000165  000020          8$:     JMP     20(R5)          ;RETURN FROM TEST
14891                                         ;
14892 074242                          MOP17:
14895                                         ;
14896                                         ;
14897                                         ;-----------------------------------
14898                                         ;TEST STCFD - USING ILLEGAL ACCUMULATOR
14899                                         ;
14900                                                 ;
14901                                                 ;
14902                                                 ;
14903 074242                          MSFDI:
14904
14905                                         ;
14906 074242  012701  040000                  MOV     #40000,R1               ;DISABLE INTERRUPTS
14907 074246  170101                          LDFPS   R1                      ;
14908 074250  176006                          STCFD   AC0,AC6                 ;*TEST ILLEGAL INSTRUCTION
14909 074252  170202                          STFPS   R2                      ;SAVE STATUS
14910 074254  170303                          STST    R3                      ;SAVE FEC
14911 074256  022702  140000                  CMP     #140000,R2              ;VERIFY FER SET
14912 074262  001401                          BEQ     1$                      ;BRANCH IF ERROR RECEIVED
14913 074264  104003                          ERROR   +3              ;FPP ERROR
14914                                                                 ;FER BIT NOT SET ON ILLEGAL INST.
14915 074266  022703  000002          1$:     CMP     #2,R3                           ;VERIFY FEC = FLOATING OPCDOE ERROR
14916 074272  001401                          BEQ     2$                      ;BRANCH IF GOOD
14917 074274  104003                          ERROR   +3              ;FPP ERROR
14918                                                                 ;FEC INCORRECT
14919 074276                          2$:
14920
14921                                         ;
14924                                         ;
14925                                         ;
14926                                         ;-----------------------------------
14927                                         ;TEST CLRD
14928                                         ;
14929                                                 ;
14930                                                 ;
14931                                                 ;
14932 074276                          MCLRD:
14933
14934                                         ;
14935 074276  012701  003764                  MOV     #TAB47,R1               ;POINT TO DATA
14936 074302  012704  000200                  MOV     #200,R4                 ;SET FPP STATUS TO DOUBLE
14937 074306  170104                          LDFPS   R4
14938 074310  172411                          LDD     (R1),AC0                ;
14939 074312  012701  003142                  MOV     #RECDST,R1              ;POINT TO DATA BUFFER
14940 074316  174011                          STD     AC0,(R1)                ;STORE GARBAGE
14941 074320  170411                          CLRD    (R1)                    ;CLEAR DATA BUFFER
14942 074322  012704  003314                  MOV     #TAB6,R4                ;VERIFY BUFFER =0
14943 074326  004767  041062                  JSR     R7,DATVER               ;
14944 074332  005767  106562                  TST     COUNT                   ;
```

FLOATING POINT TESTS

```
14945 074336  001401                        BEQ     1$                      ;BRANCH I RECDST = 0
14946 074340  104003                        ERROR   .3              ;FPP ERROR
14947                                                               ;RECDST NOT CLEARED
14948 074342  170202                1$:     STFPS   R2                      ;SAVE STATUS
14949 074344  020227  000204               CMP     R2,#204                 ;VERIFY STATUS
14950 074350  001401                        BEQ     2$                      ;BRANCH IF GOOD
14951 074352  104003                        ERROR   .3              ;FPP ERROR
14952                                                               ;BAD STATUS
14953 074354                        2$:
14954
14955
14958                               ;
14959                               ;
14960                               ;---------------------------------
14961                               ;TEST CLRD, ILLEGAL ACCUMULATOR
14962                               ;
14963                                       ;
14964                                       ;
14965                                       ;
14966 074354                        MCLRI:
14967
14968                               ;
14969 074354  012704  040200                MOV     #40200,R4               ;DISABLE INTERRUPTS
14970 074360  170104                        LDFPS   R4                      ;LOAD STATUS
14971 074362  170406                        CLRD    R6                      ;*TEST INSTRUCTION WITH ILLEGAL ACC
14972 074364  170203                        STFPS   R3                      ;SAVE STATUS
14973 074366  170305                        STST    R5                      ;SAVE FEC
14974 074370  022703  140200               CMP     #140200,R3              ;VERIFY ERROR
14975 074374  001401                        BEQ     1$                      ;BRANCH IF FER SET
14976 074376  104003                        ERROR   .3              ;FPP ERROR
14977                                                               ;ERROR IN FPS
14978 074400  022705  000002        1$:     CMP     #2,R5           ;VERIFY FEC =2 OPCODE ERROR
14979 074404  001401                        BEQ     2$              ;BRANCH IF GOOD
14980 074406  104003                        ERROR   .3              ;FPP ERROR
14981                                                                       ;BAD FEC
14982 074410                        2$:
14983
14984
14987                               ;
14988                               ;
14989                               ;---------------------------------
14990                               ;TEST LDFPS, STFPS MODE 1
14991                               ;
14992                                       ;
14993                                       ;
14994                                       ;
14995 074410                        MLS1:
14996
14997                               ;
14998 074410  012704  003162                MOV     #TSTLOC,R4              ;POINT R4 TO RAM
14999 074414  012714  147757                MOV     #147757,(R4)            ;SETUP EXPECTED STATUS
15000 074420  012701  003132                MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
15001 074424  012737  074474  000244        MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15002 074432  170114                        LDFPS   (R4)                    ;*TEST INSTRUCTION
15003 074434  170211                        STFPS   (R1)                    ;*TEST INSTRUCTION
15004 074436  020427  003162                CMP     R4,#TSTLOC              ;VERIFY R4
15005 074442  001401                        BEQ     1$                      ;BRANCH IF GOOD
```

FLOATING POINT TESTS

```
15006 074444  104003                          ERROR   .3              ;FPP ERROR
15007                                         ;
15008 074446  020127  003132          1$:     CMP     R1,#RECST               ;VERIFY R1
15009 074452  001401                          BEQ     2$                      ;BRANCH IF GOOD
15010 074454  104003                          ERROR   .3              ;FPP ERROR
15011                                         ;BAD R1
15012 074456  023727  003132  147757  2$:     CMP     @#RECST,#147757         ;VERIFY STATUS
15013 074464  001406                          BEQ     3$                      ;BRANCH F GOOD
15014 074466  104003                          ERROR   .3              ;FPP ERROR
15015                                         ;BAD STATUS\
15016 074470  000167  000006                  JMP     3$                      ;GET OVER TRAP
15017                               ;UNEXPECTED TRAP
15018 074474  012600                10$:     MOV     (SP)+,R0                        ;SAVE PC
15019 074476  012605                          MOV     (SP)+,R5                        ;SAVE PS
15020 074500  104003                          ERROR   .3              ;FPP ERROR
15021                                         ;UNEXPECTED TRAP
15022 074502                        3$:
15023
15024
15027                               ;
15028                               ;
15029                               ;---------------------------------
15030                               ;TEST LDFPS, STFPS MODE 2
15031                               ;
15032                                       ;
15033                                       ;
15034                                       ;
15035 074502                        MLS2:
15036
15037                               ;
15038 074502  012704  003162                MOV     #TSTLOC,R4              ;POINT R4 TO RAM
15039 074506  012714  145557                MOV     #145557,(R4)            ;SETUP EXPECTED STATUS
15040 074512  012701  003132                MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
15041 074516  012737  074566  000244        MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15042 074524  170124                        LDFPS   (R4)+                   ;*TEST INSTRUCTION
15043 074526  170221                        STFPS   (R1)+                   ;*TEST INSTRUCTION
15044 074530  020427  003164                CMP     R4,#TSTLOC+2            ;VERIFY R4
15045 074534  001401                        BEQ     1$                      ;BRANCH IF GOOD
15046 074536  104003                        ERROR   .3              ;FPP ERROR
15047                                       ;
15048 074540  020127  003134          1$:   CMP     R1,#RECST+2             ;VERIFY R1
15049 074544  001401                        BEQ     2$                      ;BRANCH IF GOOD
15050 074546  104003                        ERROR   .3              ;FPP ERROR
15051                                       ;BAD R1
15052 074550  023727  003132  145557  2$:   CMP     @#RECST,#145557         ;VERIFY STATUS
15053 074556  001406                        BEQ     3$                      ;BRANCH F GOOD
15054 074560  104003                        ERROR   .3              ;FPP ERROR
15055                                       ;BAD STATUS\
15056 074562  000167  000006                JMP     3$                      ;GET OVER TRAP
15057                               ;UNEXPECTED TRAP
15058 074566  012600                10$:   MOV     (SP)+,R0                        ;SAVE PC
15059 074570  012605                       MOV     (SP)+,R5                        ;SAVE PS
15060 074572  104003                       ERROR   .3              ;FPP ERROR
15061                                      ;UNEXPECTED TRAP
15062 074574                        3$:
15063
15064
```

FLOATING POINT TESTS

```
15067                                   ;
15068                                   ;
15069                                   ;---------------------------------
15070                                   ;TEST LDFPS, STFPS MODE 3
15071                                   ;
15072                                           ;
15073                                           ;
15074                                           ;
15075 074574                            MLS3:
15076
15077                                   ;
15078 074574  012704  003162                    MOV     #TSTLOC,R4              ;POINT R4 TO RAM
15079 074600  012737  003166  003162              MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
15080 074606  012737  147501  003166              MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15081 074614  012701  003172                    MOV     #TSTLOC+10,R1           ;R1 POINTS TO TSTLOC+10
15082 074620  012737  003132  003172              MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
15083 074626  012737  074676  000244              MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15084 074634  170134                            LDFPS   @(R4)+                  ;*TEST INSTRUCTION
15085 074636  170231                            STFPS   @(R1)+                  ;*TEST INSTRUCTION
15086 074640  020427  003164                    CMP     R4,#TSTLOC+2            ;VERIFY R4
15087 074644  001401                            BEQ     1$                      ;BRANCH IF GOOD
15088 074646  104003                            ERROR   +3              ;FPP ERROR
15089                                                                   ;
15090 074650  020127  003174            1$:     CMP     R1,#TSTLOC+12           ;VERIFY R1
15091 074654  001401                            BEQ     2$                      ;BRANCH IF GOOD
15092 074656  104003                            ERROR   +3              ;FPP ERROR
15093                                                                   ;BAD R1
15094 074660  023727  003132  147501    2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
15095 074666  001406                            BEQ     3$                      ;BRANCH F GOOD
15096 074670  104003                            ERROR   +3              ;FPP ERROR
15097                                                                   ;BAD STATUS\
15098 074672  000167  000006                    JMP     3$                      ;GET OVER TRAP
15099                                   ;UNEXPECTED TRAP
15100 074676  012600                    10$:    MOV     (SP)+,R0                        ;SAVE PC
15101 074700  012605                            MOV     (SP)+,R5                        ;SAVE PS
15102 074702  104003                            ERROR   +3              ;FPP ERROR
15103                                                                   ;UNEXPECTED TRAP
15104 074704                            3$:
15105
15106
15109                                   ;
15110                                   ;
15111                                   ;---------------------------------
15112                                   ;TEST LDFPS, STFPS MODE 4
15113                                   ;
15114                                           ;
15115                                           ;
15116                                           ;
15117 074704                            MLS4:
15118
15119                                   ;
15120 074704  012704  003164                    MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
15121 074710  012737  147757  003162              MOV     #147757,@#TSTLOC        ;TSTLOC= STATUS ADDRESS
15122 074716  012701  003134                    MOV     #RECST+2,R1             ;SET BUFFER FOR RECEIVED STATUS
15123 074722  012737  074772  000244              MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15124 074730  170144                            LDFPS   -(R4)                   ;*TEST INSTRUCTION
15125 074732  170241                            STFPS   -(R1)                   ;*TEST INSTRUCTION
```

```
FLOATING POINT TESTS

 15126 074734  020427  003162                  CMP     R4,#TSTLOC              ;VERIFY R4
 15127 074740  001401                          BEQ     1$                      ;BRANCH IF GOOD
 15128 074742  104003                          ERROR   .+3                   ;FPP ERROR
 15129                                                                       ;
 15130 074744  020127  003132          1$:     CMP     R1,#RECST               ;VERIFY R1
 15131 074750  001401                          BEQ     2$                      ;BRANCH IF GOOD
 15132 074752  104003                          ERROR   .+3                   ;FPP ERROR
 15133                                                                       ;BAD R1
 15134 074754  023727  003132  147757  2$:     CMP     @#RECST,#147757         ;VERIFY STATUS
 15135 074762  001406                          BEQ     3$                      ;BRANCH F GOOD
 15136 074764  104003                          ERROR   .+3                   ;FPP ERROR
 15137                                                                       ;BAD STATUS\
 15138 074766  000167  000006                  JMP     3$                      ;GET OVER TRAP
 15139                                 ;UNEXPECTED TRAP
 15140 074772  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
 15141 074774  012605                          MOV     (SP)+,R5                        ;SAVE PS
 15142 074776  104003                          ERROR   .+3                   ;FPP ERROR
 15143                                                                       ;UNEXPECTED TRAP
 15144 075000                          3$:
 15145
 15146
 15149                                 ;
 15150                                 ;
 15151                                 ;---------------------------------
 15152                                 ;TEST LDFPS, STFPS MODE 5
 15153                                 ;
 15154                                         ;
 15155                                         ;
 15156                                         ;
 15157 075000                          MLS5:
 15158
 15159                                 ;
 15160 075000  012704  003164                  MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
 15161 075004  012737  003166  003162          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
 15162 075012  012737  147501  003166          MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
 15163 075020  012701  003174                  MOV     #TSTLOC+12,R1           ;R1 POINTS TO 412
 15164 075024  012737  003132  003172          MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
 15165 075032  012737  075102  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
 15166 075040  170154                          LDFPS   @-(R4)                  ;*TEST INSTRUCTION
 15167 075042  170251                          STFPS   @-(R1)                  ;*TEST INSTRUCTION
 15168 075044  020427  003162                  CMP     R4,#TSTLOC              ;VERIFY R4
 15169 075050  001401                          BEQ     1$                      ;BRANCH IF GOOD
 15170 075052  104003                          ERROR   .+3                   ;FPP ERROR
 15171                                                                       ;
 15172 075054  020127  003172          1$:     CMP     R1,#TSTLOC+10           ;VERIFY R1
 15173 075060  001401                          BEQ     2$                      ;BRANCH IF GOOD
 15174 075062  104003                          ERROR   .+3                   ;FPP ERROR
 15175                                                                       ;BAD R1
 15176 075064  023727  003132  147501  2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
 15177 075072  001406                          BEQ     3$                      ;BRANCH F GOOD
 15178 075074  104003                          ERROR   .+3                   ;FPP ERROR
 15179                                                                       ;BAD STATUS\
 15180 075076  000167  000006                  JMP     3$                      ;GET OVER TRAP
 15181                                 ;UNEXPECTED TRAP
 15182 075102  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
 15183 075104  012605                          MOV     (SP)+,R5                        ;SAVE PS
 15184 075106  104003                          ERROR   .+3                   ;FPP ERROR
```

FLOATING POINT TESTS

```
15185                                                   ;UNEXPECTED TRAP
15186 075110                            3$:
15187
15188
15191                                   ;
15192                                   ;
15193                                   ;---------------------------------
15194                                   ;TEST LDFPS, STFPS MODE 6
15195                                   ;
15196                                           ;
15197                                           ;
15198                                           ;
15199 075110                            MLS6:
15200
15201                                   ;
15202 075110  012704  003162                      MOV     #TSTLOC,R4              ;POINT R4 TO RAM
15203 075114  012737  140001  003166              MOV     #140001,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15204 075122  012701  003272                      MOV     #TSTLOC+110,R1          ;R1 WILL POINT TO TESTLOC+10
15205 075126  012737  075202  000244              MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15206 075134  170164  000004                      LDFPS   4(R4)                   ;*TEST INSTRUCTION
15207 075140  170261  177700                      STFPS   -100(R1)                        ;*TEST INSTRUCTION
15208 075144  020427  003162                      CMP     R4,#TSTLOC              ;VERIFY R4
15209 075150  001401                              BEQ     1$                      ;BRANCH IF GOOD
15210 075152  104003                              ERROR   +3              ;FPP ERROR
15211                                                                   ;
15212 075154  020127  003272              1$:     CMP     R1,#TSTLOC+110          ;VERIFY R1
15213 075160  001401                              BEQ     2$                      ;BRANCH IF GOOD
15214 075162  104003                              ERROR   +3              ;FPP ERROR
15215                                                                   ;BAD R1
15216 075164  023727  003172  140001      2$:     CMP     @#TSTLOC+10,#140001     ;VERIFY STATUS
15217 075172  001406                              BEQ     3$                      ;BRANCH F GOOD
15218 075174  104003                              ERROR   +3              ;FPP ERROR
15219                                                                   ;BAD STATUS\
15220 075176  000167  000006                      JMP     3$                      ;GET OVER TRAP
15221                                   ;UNEXPECTED TRAP
15222 075202  012600                    10$:    MOV     (SP)+,R0                        ;SAVE PC
15223 075204  012605                            MOV     (SP)+,R5                        ;SAVE PS
15224 075206  104003                            ERROR   +3              ;FPP ERROR
15225                                                                   ;UNEXPECTED TRAP
15226 075210                            3$:
15227
15228
15231                                   ;
15232                                   ;
15233                                   ;---------------------------------
15234                                   ;TEST LDFPS, STFPS MODE 7
15235                                   ;
15236                                           ;
15237                                           ;
15238                                           ;
15239 075210                            MLS7:
15240
15241                                   ;
15242 075210  012704  003262                      MOV     #TSTLOC+100,R4          ;POINT R4 TO RAM
15243 075214  012737  003166  003162              MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
15244 075222  012737  145501  003166              MOV     #145501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15245 075230  012701  003072                      MOV     #TSTLOC-70,R1           ;R1 POINTS TO TSTLOC+10
```

FLOATING POINT TESTS

```
15246 075234  012737  003172  003164          MOV     @TSTLOC+10,@@TSTLOC+2   ;
15247 075242  012737  075316  000244          MOV     #10$,@@FPVEC            ;SETUP TRAP VECTOR
15248 075250  170174  177700                  LDFPS   @-100(R4)                       ;*TEST INSTRUCTION
15249 075254  170271  000072                  STFPS   @72(R1)                 ;*TEST INSTRUCTION
15250 075260  020427  003262                  CMP     R4,@TSTLOC+100          ;VERIFY R4
15251 075264  001401                          BEQ     1$                      ;BRANCH IF GOOD
15252 075266  104003                          ERROR   +3              ;FPP ERROR
15253                                                                 ;
15254 075270  020127  003072          1$:     CMP     R1,@TSTLOC-70           ;VERIFY R1
15255 075274  001401                          BEQ     2$                      ;BRANCH IF GOOD
15256 075276  104003                          ERROR   +3              ;FPP ERROR
15257                                                                 ;BAD R1
15258 075300  023727  003172  145501  2$:     CMP     @@TSTLOC+10,#145501     ;VERIFY STATUS
15259 075306  001406                          BEQ     3$                      ;BRANCH F GOOD
15260 075310  104003                          ERROR   +3              ;FPP ERROR
15261                                                                 ;BAD STATUS\
15262 075312  000167  000006                  JMP     3$                      ;GET OVER TRAP
15263                                 ;UNEXPECTED TRAP
15264 075316  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15265 075320  012605                          MOV     (SP)+,R5                        ;SAVE PS
15266 075322  104003                          ERROR   +3              ;FPP ERROR
15267                                                                 ;UNEXPECTED TRAP
15268 075324                          3$:
15269
15270
15273                                 ;
15274                                 ;
15275                                 ;---------------------------------
15276                                 ;TEST LDCLD MODE 27
15277                                 ;
15278                                         ;
15279                                         ;
15280                                         ;
15281 075324                          MLDC2:
15282
15283                                 ;
15284 075324  005001                          CLR     R1                      ;INIT R1
15285 075326  012704  007700                  MOV     #7700,R4                ;FPS=DOUBLE, LONG
15286 075332  170104                          LDFPS   R4                      ;
15287 075334  012737  075370  000244          MOV     #10$,@@FPVEC            ;SETUP WILD TRAP
15288 075342  177027                          LDCLD   (R7)+,AC0               ;*TEST INSTRUCTION
15289 075344  005201                          INC     R1                      ;
15290 075346  005201                          INC     R1                      ;
15291 075350  005201                          INC     R1                      ;
15292 075352  005201                          INC     R1                      ;
15293 075354  020127  000003                  CMP     R1,#3                   ;VERIFY
15294 075360  001406                          BEQ     1$                      ;BRANCH IF GOOD
15295 075362  104003                          ERROR   +3              ;FPP ERROR
15296                                                                 ;INSTRUCTION FAILED
15297 075364  000167  000006                  JMP     1$                      ;JUMP OVER WILD TRAP
15298 075370  012600                  10$:    MOV     (SP)+,R0                ;SAVE PC
15299 075372  012605                          MOV     (SP)+,R5                        ;SAVE PS
15300 075374  104003                          ERROR   +3              ;FPP ERROR
15301                                                                 ;WILD TRAP ON INSTRUCTION
15302 075376  012704  003324          1$:     MOV     @TAB6A,R4               ;POINT TO EXPECTED DATA
15303 075402  012701  003142                  MOV     @RECDST,R1              ;POINT TO DATA BUFFER
15304 075406  174011                          STD     AC0,(R1)                ;VERIFY DATA
```

FLOATING POINT TESTS

```
15305 075410  004767  040000                  JSR     R7,DATVER               ;
15306 075414  005767  105500                  TST     COUNT
15307 075420  001401                          BEQ     2$                      ;BRANCH IF GOOD DATA
15308 075422  104003                          ERROR   +3              ;FPP ERROR
15309                                                                 ;BAD DATA
15310 075424                          2$:
15311
15312
15313
15316                                 ;
15317                                 ;
15318                                 ;---------------------------------
15319                                 ;TEST LDCIF, LDCLF
15320                                 ;
15321                                         ;
15322                                         ;
15323                                         ;
15324 075424                          MLCF:
15325
15326                                 ;
15327                                 ;1/INT=0
15328 075424  004767  000500                  JSR     R7,LCFSUB               ;DO TEST
15329 075430  000000  000000                  .WORD   0.0             ;FSRC
15330 075434  000000  000000                  .WORD   0.0             ;RESULT
15331 075440  000000                          .WORD   0               ; TEST FPS
15332 075442  000004                          .WORD   4               ;RESULT FPS
15333                                 ;2/INT=0,-1
15334 075444  004767  000460                  JSR     R7,LCFSUB               ;DO TEST
15335 075450  000000  177777                  .WORD   0,-1            ;FSRC
15336 075454  000000  000000                  .WORD   0.0             ;RESULT
15337 075460  007440                          .WORD   7440            ; TEST FPS
15338 075462  007444                          .WORD   7444            ;RESULT FPS
15339                                 ;3/LONG=0
15340 075464  004767  000440                  JSR     R7,LCFSUB               ;DO TEST
15341 075470  000000  000000                  .WORD   0.0             ;FSRC
15342 075474  000000  000000                  .WORD   0.0             ;RESULT
15343 075500  000100                          .WORD   100             ; TEST FPS
15344 075502  000104                          .WORD   104             ;RESULT FPS
15345                                 ;4/INT=40000
15346 075504  004767  000420                  JSR     R7,LCFSUB               ;DO TEST
15347 075510  040000  000000                  .WORD   40000.0         ;FSRC
15348 075514  043600  000000                  .WORD   43600.0         ;RESULT
15349 075520  000017                          .WORD   17              ; TEST FPS
15350 075522  000000                          .WORD   0               ;RESULT FPS
15351                                 ;5/LONG=1
15352 075524  004767  000400                  JSR     R7,LCFSUB               ;DO TEST
15353 075530  000000  000001                  .WORD   0,1             ;FSRC
15354 075534  040200  000000                  .WORD   40200.0         ;RESULT
15355 075540  000117                          .WORD   117             ; TEST FPS
15356 075542  000100                          .WORD   100             ;RESULT FPS
15357                                 ;6/INT=PATTERN
15358 075544  004767  000360                  JSR     R7,LCFSUB               ;DO TEST
15359 075550  000252  025252                  .WORD   252,25252       ;FSRC
15360 075554  042052  000000                  .WORD   42052.0         ;RESULT
15361 075560  000000                          .WORD   0               ; TEST FPS
15362 075562  000000                          .WORD   0               ;RESULT FPS
15363                                 ;7/INT=-40000
```

FLOATING POINT TESTS

```
15364 075564  004767  000340                JSR     R7,LCFSUB               ;DO TEST
15365 075570  140000  000000                .WORD   -40000,0        ;FSRC
15366 075574  143600  000000                .WORD   143600,0                ;RESULT
15367 075600  000007                        .WORD   7               ; TEST FPS
15368 075602  000010                        .WORD   10              ;RESULT FPS
15369                               ;8/INT=-1
15370 075604  004767  000320                JSR     R7,LCFSUB               ;DO TEST
15371 075610  177777  000000                .WORD   -1,0            ;FSRC
15372 075614  140200  000000                .WORD   140200,0                ;RESULT
15373 075620  000007                        .WORD   7               ; TEST FPS
15374 075622  000010                        .WORD   10              ;RESULT FPS
15375                               ;9/INT=PATTERN
15376 075624  004767  000300                JSR     R7,LCFSUB               ;DO TEST
15377 075630  125252  125252                .WORD   125252,125252           ;FSRC
15378 075634  143652  126000                .WORD   143652,126000           ;RESULT
15379 075640  000007                        .WORD   7               ; TEST FPS
15380 075642  000010                        .WORD   10              ;RESULT FPS
15381                               ;10/LONG=40000
15382 075644  004767  000260                JSR     R7,LCFSUB               ;DO TEST
15383 075650  040000  000000                .WORD   40000,0         ;FSRC
15384 075654  047600  000000                .WORD   47600,0         ;RESULT
15385 075660  000117                        .WORD   117             ; TEST FPS
15386 075662  000100                        .WORD   100             ;RESULT FPS
15387                               ;11/LONG=1
15388 075664  004767  000240                JSR     R7,LCFSUB               ;DO TEST
15389 075670  000000  000001                .WORD   0,1             ;FSRC
15390 075674  040200  000000                .WORD   40200,0         ;RESULT
15391 075700  007557                        .WORD   7557            ; TEST FPS
15392 075702  007540                        .WORD   7540            ;RESULT FPS
15393                               ;12/LONG=PATTERN
15394 075704  004767  000220                JSR     R7,LCFSUB               ;DO TEST
15395 075710  000000  000252                .WORD   0,252           ;FSRC
15396 075714  042052  000000                .WORD   42052,0         ;RESULT
15397 075720  007557                        .WORD   7557            ; TEST FPS
15398 075722  007540                        .WORD   7540            ;RESULT FPS
15399                               ;13/LONG = -40000
15400 075724  004767  000200                JSR     R7,LCFSUB               ;DO TEST
15401 075730  140000  000000                .WORD   -40000,0                ;FSRC
15402 075734  147600  000000                .WORD   147600,0                ;RESULT
15403 075740  000107                        .WORD   107             ; TEST FPS
15404 075742  000110                        .WORD   110             ;RESULT FPS
15405                               ;14/LONG=-1
15406 075744  004767  000160                JSR     R7,LCFSUB               ;DO TEST
15407 075750  177777  177777                .WORD   -1,-1           ;FSRC
15408 075754  140200  000000                .WORD   140200,0                ;RESULT
15409 075760  007500                        .WORD   7500            ; TEST FPS
15410 075762  007510                        .WORD   7510            ;RESULT FPS
15411                               ;15/LONG=PATTERN
15412 075764  004767  000140                JSR     R7,LCFSUB               ;DO TEST
15413 075770  125252  125252                .WORD   125252,125252           ;FSRC
15414 075774  147652  125253                .WORD   147652,125253           ;RESULT
15415 076000  000105                        .WORD   105             ; TEST FPS
15416 076002  000110                        .WORD   110             ;RESULT FPS
15417                               ;16/LONG=77777,177500
15418 076004  004767  000120                JSR     R7,LCFSUB               ;DO TEST
15419 076010  077777  177500                .WORD   77777,177500            ;FSRC
15420 076014  047777  177777                .WORD   47777,177777            ;RESULT
```

FLOATING POINT TESTS

```
15421 076020  000117                    .WORD   117             ; TEST FPS
15422 076022  000100                    .WORD   100             ;RESULT FPS
15423                           ;17/LONG=40000,100
15424 076024  004767  000100            JSR     R7,LCFSUB               ;DO TEST
15425 076030  040000  000100            .WORD   40000,100               ;FSRC
15426 076034  047600  000001            .WORD   47600,1         ;RESULT
15427 076040  007502                    .WORD   7502            ; TEST FPS
15428 076042  007500                    .WORD   7500            ;RESULT FPS
15429                           ;18/LONG=40000,100 - TRUNCATE
15430 076044  004767  000060            JSR     R7,LCFSUB               ;DO TEST
15431 076050  040000  000100            .WORD   40000,100               ;FSRC
15432 076054  047600  000000            .WORD   47600,0         ;RESULT
15433 076060  007557                    .WORD   7557            ; TEST FPS
15434 076062  007540                    .WORD   7540            ;RESULT FPS
15435                           ;19/INT= MOST NEGATIVE
15436 076064  004767  000040            JSR     R7,LCFSUB               ;DO TEST
15437 076070  100000  000000            .WORD   100000,0                ;FSRC
15438 076074  144000  000000            .WORD   144000,0                ;RESULT
15439 076100  000007                    .WORD   7               ; TEST FPS
15440 076102  000010                    .WORD   10              ;RESULT FPS
15441                           ;20/LONG= MOST NEGATIVE
15442 076104  004767  000020            JSR     R7,LCFSUB               ;DO TEST
15443 076110  100000  000000            .WORD   100000,0                ;FSRC
15444 076114  150000  000000            .WORD   150000,0                ;RESULT
15445 076120  000107                    .WORD   107             ; TEST FPS
15446 076122  000110                    .WORD   110             ;RESULT FPS
15447                           ;
15448
15449 076124  000167  000112            JMP     MOP18           ;GET OVER SUBROUTINE
15450                           ;
15451                           ;
15452                           ;*X*X*X
```

FLOATING POINT TESTS

```
15478 076166  174011                          STD     AC0,(R1)                ;SAVE TEST RESULT INTO RECDST
15479 076170  016200  000012                  MOV     12(R2),R0                       ;GET EXPECTED STATUS
15480 076174  020003                          CMP     R0,R3                   ;VERIFY STATUS
15481 076176  001401                          BEQ     3$                      ;BRANCH IF GOOD
15482 076200  104003                          ERROR   .3                      ;FPP ERROR
15483                                                                         ;BAD FPS
15484 076202  010204                  3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
15485 076204  062704  000004                  ADD     #4,R4
15486 076210  004767  037162          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
15487 076214  005767  104700                  TST     COUNT
15488 076220  001401                          BEQ     5$                      ;BRANCH IF GOOD
15489 076222  104003                          ERROR   .3                      ;FPP ERROR
15490                                                                         ;BAD AC0
15491 076224  000162  000014          5$:     JMP     14(R2)                  ;RETURN FROM TEST
15492                                         ;
15493                                         ;INSTRUCTION TRAPPED
15494 076230  012600                  50$:    MOV     (SP)+,R0                ;SAVE PC
15495 076232  012605                          MOV     (SP)+,R5                ;SAVE PS
15496 076234  104003                          ERROR   .3                      ;FPP ERROR
15497                                                                         ;INSTRUCTION WASNT SUPPOSE TO TRAP
15498 076236  000167  177762                  JMP     5$                      ;CONTINUE
15499 076242                          MCP18:
15500
15503                                         ;
15504                                         ;
15505                                         ;---------------------------------
15506                                         ;TEST LDCID, LDCLD
15507                                         ;
15508                                                 ;
15509                                                 ;
15510                                                 ;
15511 076242                          MLCD:
15512
15513                                         ;
15514                                         ;1/LONG=0
15515 076242  004767  000264                          JSR     R7,LCDSUB                       ;DO TEST
15516 076246  000000  000000                          .WORD   0,0                     ;FSRC
15517 076252  000000  000000  000000                  .WORD   0,0,0,0                 ;RESULT
      076260  000000
15518 076262  007313                                  .WORD   7313                    ; TEST FPS
15519 076264  007304                                  .WORD   7304                    ;RESULT FPS
15520                                         ;2/INT=0
15521 076266  004767  000240                          JSR     R7,LCDSUB                       ;DO TEST
15522 076272  000000  000001                          .WORD   0,1                     ;FSRC
15523 076276  040200  000000  000000                  .WORD   40200,0,0,0                     ;RESULT
      076304  000000
15524 076306  007757                                  .WORD   7757                    ; TEST FPS
15525 076310  007740                                  .WORD   7740                    ;RESULT FPS
15526                                         ;3/INT=40000
15527 076312  004767  000214                          JSR     R7,LCDSUB                       ;DO TEST
15528 076316  040000  177777                          .WORD   40000,-1                        ;FSRC
15529 076322  043600  000000  000000                  .WORD   43600,0,0,0                     ;RESULT
      076330  000000
15530 076332  007617                                  .WORD   7617                    ; TEST FPS
15531 076334  007600                                  .WORD   7600                    ;RESULT FPS
15532                                         ;4/INT=-40000
15533 076336  004767  000170                          JSR     R7,LCDSUB                       ;DO TEST
```

FLOATING POINT TESTS

```
15534 076342  140000  177777                .WORD   -40000,-1               ;FSRC
15535 076346  143600  000000  000000        .WORD   143600,0,0,0            ;RESULT
      076354  000000
15536 076356  007600                        .WORD   7600            ; TEST FPS
15537 076360  007610                        .WORD   7610            ;RESULT FPS
15538                             ;5/LONG=40000
15539 076362  004767  000144                JSR     R7,LCDSUB               ;DO TEST
15540 076366  040000  000000                .WORD   40000,0         ;FSRC
15541 076372  047600  000000  000000        .WORD   47600,0,0,0             ;RESULT
      076400  000000
15542 076402  007757                        .WORD   7757            ; TEST FPS
15543 076404  007740                        .WORD   7740            ;RESULT FPS
15544                             ;6/LONG=1
15545 076406  004767  000120                JSR     R7,LCDSUB               ;DO TEST
15546 076412  000000  000001                .WORD   0,1             ;FSRC
15547 076416  040200  000000  000000        .WORD   40200,0,0,0             ;RESULT
      076424  000000
15548 076426  000300                        .WORD   300             ; TEST FPS
15549 076430  000300                        .WORD   300             ;RESULT FPS
15550                             ;7/LONG=-2
15551 076432  004767  000074                JSR     R7,LCDSUB               ;DO TEST
15552 076436  177777  177776                .WORD   -1,-2           ;FSRC
15553 076442  140400  000000  000000        .WORD   140400,0,0,0            ;RESULT
      076450  000000
15554 076452  007300                        .WORD   7300            ; TEST FPS
15555 076454  007310                        .WORD   7310            ;RESULT FPS
15556                             ;8/INT=PATTERN
15557 076456  004767  000050                JSR     R7,LCDSUB               ;DO TEST
15558 076462  123456  176543                .WORD   123456,176543           ;FSRC
15559 076466  143661  122000  000000        .WORD   143661,122000,0,0               ;RESULT
      076474  000000
15560 076476  000200                        .WORD   200             ; TEST FPS
15561 076500  000210                        .WORD   210             ;RESULT FPS
15562                             ;9/LONG=PATTERN
15563 076502  004767  000024                JSR     R7,LCDSUB               ;DO TEST
15564 076506  125252  125252                .WORD   125252,125252           ;FSRC
15565 076512  147652  125252  126000        .WORD   147652,125252,126000,0          ;RESULT
      076520  000000
15566 076522  000300                        .WORD   300             ; TEST FPS
15567 076524  000310                        .WORD   310             ;RESULT FPS
15568                             ;
15569
15570                             ;
15571 076526  000167  000112                JMP     HOP19           ;GET OVER SUBROUTINE
15572                             ;
15573                             ;*X*X*X
```

F6

FLOATING POINT TESTS

```
15585                                           ;
15586 076532  012602                    LCDSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
15587 076534  012737  076632  000244            MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
15588 076542  012701  003142                    MOV     #RECDST,R1              ;POINT TO RESULT AREA
15589 076546  016200  000014                    MOV     14(R2),R0               ;GET TEST FPS
15590 076552  170100                            LDFPS   R0                      ;LOAD TEST FPS
15591 076554  010204                            MOV     R2,R4                   ;POINT TO TEST DATA
15592                                           ;
15593 076556  177014                    40$:    LDCID   (R4),AC0                ;*TEST INSTRUCTION (ACCORDING TO MODE)
15594                                           ;
15595                                   ;VERIFY STATUS
15596 076560  170203                    2$:     STFPS   R3                      ;SAVE FPS
15597 076562  012700  000200                    MOV     #200,R0                 ;SET FPP STATUS TO DOUBLE
15598 076566  170100                            LDFPS   R0                      ;
15599 076570  174011                            STD     AC0,(R1)                ;SAVE TEST RESULT INTO RECDST
15600 076572  016200  000016                    MOV     16(R2),R0                       ;GET EXPECTED STATUS
15601 076576  020003                            CMP     R0,R3                   ;VERIFY STATUS
15602 076600  001401                            BEQ     3$                      ;BRANCH IF GOOD
15603 076602  104003                            ERRCR   .+3             ;FPP ERROR
15604                                           ;BAD FPS
15605 076604  010204                    3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
15606 076606  062704  000004                    ADD     #4,R4
15607 076612  004767  036576            4$:     JSR     R7,DATVER                       ;VERIFY DATA
15608 076616  005767  104276                    TST     COUNT
15609 076622  001401                            BEQ     5$                      ;BRANCH IF GOOD
15610 076624  104003                            ERROR   .+3             ;FPP ERROR
15611                                           ;BAD AC0
15612 076626  000162  000020            5$:     JMP     20(R2)                  ;RETURN FROM TEST
15613                                           ;
15614                                   ;INSTRUCTION TRAPPED
15615 076632  012600                    50$:    MOV     (SP)+,R0                ;SAVE PC
15616 076634  012605                            MOV     (SP)+,R5                ;SAVE PS
15617 076636  104003                            ERROR   .+3             ;FPP ERROR
15618                                           ;INSTRUCTION WASNT SUPPOSE TO TRAP
15619 076640  000167  177762                    JMP     5$                      ;CONTINUE
15620                                           ;
15621 076644                            MOP19:
15622
15625                                           ;
15626                                           ;
15627                                           ;---------------------------------
15628                                           ;TEST LDEXP
15629                                           ;DOUBLE
15630                                           ;
15631                                                   ;
15632                                                   ;
15633                                                   ;
15634 076644                            MLXP:
15635
15636                                           ;
15637                                           ;1/EXP=10 - AC=NEG
15638 076644  005037  003030                            CLR     @#FLAG                  ;NO INTERRUPTS
15639 076650  004767  001140                            JSR     R7,LXPSUB               ;DO TEST
15640 076654  123456  067012  025252                    .WORD   123456,67012,25252,171717       ;AC0
      076662  171717
15641 076664  000010                                    .WORD   10                      ;EXP
15642 076666  142056  067012  025252                    .WORD   142056,67012,25252,171717               ;RESULT
```

FLOATING POINT TESTS

```
        076674  171717
15643   076676  007757                          .WORD   7757            ; TEST FPS
15644   076700  007750                          .WORD   7750            ;RESULT FPS
15645                           ;2/EXP=177 - ACO=POS
15646   076702  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15647   076706  004767  001102                  JSR     R7,LXPSUB               ;DO TEST
15648   076712  023456  070123  100000          .WORD   23456,70123,100000,1            ;ACO
        076720  000001
15649   076722  000177                          .WORD   177             ;EXP
15650   076724  077656  070123  100000          .WORD   77656,70123,100000,1            ;RESULT
        076732  000001
15651   076734  007700                          .WORD   7700            ; TEST FPS
15652   076736  007700                          .WORD   7700            ;RESULT FPS
15653                           ;3/EXP=56
15654   076740  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15655   076744  004767  001044                  JSR     R7,LXPSUB               ;DO TEST
15656   076750  055555  044444  033333          .WORD   55555,44444,33333,22222         ;ACO
        076756  022222
15657   076760  000056                          .WORD   56              ;EXP
15658   076762  053555  044444  033333          .WORD   53555,44444,33333,22222 ;RESULT
        076770  022222
15659   076772  007757                          .WORD   7757            ; TEST FPS
15660   076774  007740                          .WORD   7740            ;RESULT FPS
15661                           ;4/EXP=-151, ACO=UV
15662   076776  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15663   077002  004767  001006                  JSR     R7,LXPSUB               ;DO TEST
15664   077006  100077  177777  177777          .WORD   100077,-1,-1.-2         ;ACO
        077014  177776
15665   077016  177623                          .WORD   -155            ;EXP
15666   077020  104677  177777  177777          .WORD   104677,-1,-1,-2         ;RESULT
        077026  177776
15667   077030  007757                          .WORD   7757            ; TEST FPS
15668   077032  007750                          .WORD   7750            ;RESULT FPS
15669                           ;5/EXP=-177
15670   077034  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15671   077040  004767  000750                  JSR     R7,LXPSUB               ;DO TEST
15672   077044  000177  177777  177777          .WORD   177,-1,-1,-2            ;ACO
        077052  177776
15673   077054  177601                          .WORD   -177            ;EXP
15674   077056  000377  177777  177777          .WORD   377,-1,-1,-2            ;RESULT
        077064  177776
15675   077066  007700                          .WORD   7700            ; TEST FPS
15676   077070  007700                          .WORD   7700            ;RESULT FPS
15677                           ;6/EXP=-200, UNDERFLOW
15678   077072  012737  000001  003030          MOV     #1,@#FLAG               ; INTERRUPTS
15679   077100  004767  000710                  JSR     R7,LXPSUB               ;DO TEST
15680   077104  030131  032334  035363          .WORD   30131,32334,35363,73031         ;ACO
        077112  073031
15681   077114  177600                          .WORD   -200            ;EXP
15682   077116  000131  032334  035363          .WORD   131,32334,35363,73031           ;RESULT
        077124  073031
15683   077126  007740                          .WORD   7740            ; TEST FPS
15684   077130  107744                          .WORD   107744          ;RESULT FPS
15685   077132  000012                          .WORD   12              ;FEC
15686                           ;7/EXP=LARGEST NEGATIVE
15687   077134  012737  000001  003030          MOV     #1,@#FLAG               ;EXPECT  INTERRUPTS
15688   077142  004767  000646                  JSR     R7,LXPSUB               ;DO TEST
```

FLOATING POINT TESTS

```
15689 077146  000000  000123  000456          .WORD   0,123,456,1             ;AC0
      077154  000001
15690 077156  100000                          .WORD   100000          ;EXP
15691 077160  040000  000123  000456          .WORD   40000,123,456,1         ;RESULT
      077166  000001
15692 077170  002200                          .WORD   2200            ; TEST FPS
15693 077172  102200                          .WORD   102200          ;RESULT FPS
15694 077174  000012                          .WORD   12              ;FEC
15695                               ;8/EXP=-200, NEG. AC0
15696 077176  012737  000001  003030          MOV     #1,@#FLAG               ; INTERRUPTS
15697 077204  004767  000604                  JSR     R7,LXPSUB               ;DO TEST
15698 077210  111111  100000  100000          .WORD   111111,100000,100000,-1         ;AC0
      077216  177777
15699 077220  177600                          .WORD   -200            ;EXP
15700 077222  100111  100000  100000          .WORD   100111,100000,100000,-1         ;RESULT
      077230  177777
15701 077232  002217                          .WORD   2217            ; TEST FPS
15702 077234  102214                          .WORD   102214          ;RESULT FPS
15703 077236  000012                          .WORD   12              ;FEC
15704                               ;9/EXP=-1743, FIU=0
15705 077240  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPTS
15706 077246  004767  000542                  JSR     R7,LXPSUB               ;DO TEST
15707 077252  123456  012346  012346          .WORD   123456,12346,12346,123          ;AC0
      077260  000123
15708 077262  176035                          .WORD   -1743           ;EXP
15709 077264  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
      077272  000000
15710 077274  005700                          .WORD   5700            ; TEST FPS
15711 077276  005704                          .WORD   5704            ;RESULT FPS
15712 077300  000012                          .WORD   12              ;FEC
15713                               ;10/EXP =-16616, FID=1
15714 077302  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPTS
15715 077310  004767  000500                  JSR     R7,LXPSUB               ;DO TEST
15716 077314  000377  123456  065432          .WORD   377,123456,65432,1              ;AC0
      077322  000001
15717 077324  161162                          .WORD   -16616          ;EXP
15718 077326  074577  123456  065432          .WORD   74577,123456,65432,1            ;RESULT
      077334  000001
15719 077336  047700                          .WORD   47700           ; TEST FPS
15720 077340  147700                          .WORD   147700          ;RESULT FPS
15721 077342  000012                          .WORD   12              ;FEC
15722                               ;11/EXP=177, AC0=UNDEFINED VARIABLE
15723 077344  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
15724 077350  004767  000440                  JSR     R7,LXPSUB               ;DO TEST
15725 077354  100177  177777  177777          .WORD   100177,-1,-1,-1         ;AC0
      077362  177777
15726 077364  000177                          .WORD   177             ;EXP
15727 077366  177777  177777  177777          .WORD   -1,-1,-1,-1             ;RESULT
      077374  177777
15728 077376  007700                          .WORD   7700            ; TEST FPS
15729 077400  007710                          .WORD   7710            ;RESULT FPS
15730                               ;12/EXP=150 AC0=POS
15731 077402  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
15732 077406  004767  000402                  JSR     R7,LXPSUB               ;DO TEST
15733 077412  000200  000100  000200          .WORD   200,100,200,300         ;AC0
      077420  000300
15734 077422  000150                          .WORD   150             ;EXP
```

FLOATING POINT TESTS

```
15735 077424  072000  000100  000200          .WORD   72000,100,200,300               ;RESULT
      077432  000300
15736 077434  007717                          .WORD   7717            ; TEST FPS
15737 077436  007700                          .WORD   7700            ;RESULT FPS
15738                               ;13/EXP=200, AC0=NEG
15739 077440  012737  000001  003030          MOV     #1,@#FLAG
15740 077446  004767  000342                  JSR     R7,LXPSUB               ;DO TEST
15741 077452  177777  177777  177777          .WORD   -1,-1,-1,-1             ;AC0
      077460  177777
15742 077462  000200                          .WORD   200             ;EXP
15743 077464  100177  177777  177777          .WORD   100177,-1,-1,-1         ;RESULT
      077472  177777
15744 077474  007705                          .WORD   7705            ; TEST FPS
15745 077476  107716                          .WORD   107716          ;RESULT FPS
15746 077500  000010                          .WORD   10              ;FEC
15747                               ;14/EXP=400, FID
15748 077502  012737  000002  003030          MOV     #2,@#FLAG               ;INTERRUPT
15749 077510  004767  000300                  JSR     R7,LXPSUB               ;DO TEST
15750 077514  000555  177777  177776          .WORD   555,-1,-2,-3            ;AC0
      077522  177775
15751 077524  000400                          .WORD   400                     ;EXP
15752 077526  040155  177777  177776          .WORD   40155,-1,-2,-3          ;RESULT
      077534  177775
15753 077536  047700                          .WORD   47700           ; TEST FPS
15754 077540  147702                          .WORD   147702          ;RESULT FPS
15755 077542  000010                          .WORD   10              ;FEC
15756                               ;15/EXP=11011 FIU=0
15757 077544  012737  000000  003030          MOV     #0,@#FLAG               ;NO INTERRUPT
15758 077552  004767  000236                  JSR     R7,LXPSUB               ;DO TEST
15759 077556  177773  177777  177776          .WORD   177773,-1,-2,-3         ;AC0
      077564  177775
15760 077566  011011                          .WORD   11011           ;EXP
15761 077570  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
      077576  000000
15762 077600  006700                          .WORD   6700            ; TEST FPS
15763 077602  006706                          .WORD   6706            ;RESULT FPS
15764                               ;16/EXP=LARGEST POSITIVE
15765 077604  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
15766 077612  004767  000176                  JSR     R7,LXPSUB               ;DO TEST
15767 077616  123456  000100  000100          .WORD   123456,100,100,200              ;AC0
      077624  000200
15768 077626  077777                          .WORD   77777           ;EXP
15769 077630  137656  000100  000100          .WORD   137656,100,100,200              ;RESULT
      077636  000200
15770 077640  007740                          .WORD   7740            ; TEST FPS
15771 077642  107752                          .WORD   107752          ;RESULT FPS
15772 077644  000010                          .WORD   10              ;FEC
15773                               ;17/FLOATING
15774 077646  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
15775 077652  004767  000136                  JSR     R7,LXPSUB               ;DO TEST
15776 077656  123456  023465  000555          .WORD   123456,23465,555,444            ;AC0
      077664  000444
15777 077666  000050                          .WORD   50              ;EXP
15778 077670  152056  023465  000555          .WORD   152056,23465,555,444            ;RESULT
      077676  000444
15779 077700  007500                          .WORD   7500            ; TEST FPS
15780 077702  007510                          .WORD   7510            ;RESULT FPS
```

```
                                        ;18/FLOATING UNDERFLOW
077704  012737  000001  003030                  MOV     #1,@#FLAG                       ;INTERRUPT
077712  004767  000076                          JSR     R7,LXPSUB                       ;DO TEST
077716  000333  000444  000555                  .WORD   333,444,555,666                 ;AC0
077724  000666
077726  177600                                  .WORD   -200                    ;EXP
077730  000133  000444  000555                  .WORD   133,444,555,666                 ;RESULT
077736  000666
077740  007500                                  .WORD   7500                    ; TEST FPS
077742  107504                                  .WORD   107504                  ;RESULT FPS
077744  000012                                  .WORD   12                      ;FEC
                                        ;19/FLOATING OVERFLOW
077746  012737  000001  003030                  MOV     #1,@#FLAG                       ;INTERRUPT
077754  004767  000034                          JSR     R7,LXPSUB                       ;DO TEST
077760  012346  000123  000345                  .WORD   12346,123,345,456                       ;AC0
077766  000456
077770  000400                                  .WORD   400                     ;EXP
077772  040146  000123  000345                  .WORD   40146,123,345,456                       ;RESULT
100000  000456
100002  007400                                  .WORD   7400                    ; TEST FPS
100004  107402                                  .WORD   107402                  ;RESULT FPS
100006  000010                                  .WORD   10                      ;FEC
                                        ;
                                        ;
100010  000167  000220                          JMP     HOP20                           ;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
                                        ;LDEXP
                                        ;               AC0
                                        ;               EXPONENT
                                        ;               RESULT
                                        ;               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*X*X
                                        ;
100014  012602                          LXPSUB: MOV     (SP)+,R2                        ; RETURN ADDRESS TO USE AS POINTER
100016  012737  100100  000244                  MOV     #50$,@#FPVEC                    ;REDIRECT TRAP VECTOR
100024  012701  003142                          MOV     #RECDST,R1                      ;POINT TO RESULT AREA
100030  012700  000200                          MOV     #200,R0                         ;SET FPS TO DOUBLE
100034  170100                                  LDFPS   R0                              ;
100036  010204                                  MOV     R2,R4                           ;POINT TO AC0 DATA
100040  172414                                  LDD     (R4),AC0                                ;LOAD AC0
100042  016200  000022                          MOV     22(R2),R0                       ;GET TEST FPS
100046  170100                                  LDFPS   R0                              ;LOAD TEST FPS
100050  016204  000010                          MOV     10(R2),R4                       ;POINT TO TEST DATA
                                        ;
100054  176404                          40$:    LDEXP   R4,AC0                          ;*TEST INSTRUCTION (ACCORDING TO MODE)
100056  170327                          1$:     STST    (PC)+                           ;WAIT FOR POSSIBLE FPA TRAP.
100060  000000                                  .WORD   0                               ;STORE STATUS HERE
                                        ;
                                        ;
                                        ;INSTRUCTION DIDNT TRAP
100062  032737  000001  003030                  BIT     #1,@#FLAG                               ;VERIFY A NO TRAP CONDITION
100070  001420                                  BEQ     2$                              ;BRANCH IF GOOD
```

FLOATING POINT TESTS

```
15834 100072  104003                          ERROR   +3              ;FPP ERROR
15835                                                                 ;INSTRUCTION SHOULD HAVE TRAPPED
15836 100074  000167  000032                  JMP     2$                      ;REJOIN CODE
15837                                 ;
15838                                 ;INSTRUCTION TRAPPED
15839 100100  032737  000001  003030  50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
15840 100106  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
15841 100110  104003                          ERROR   +3              ;FPP ERROR
15842                                                                 ;INSTRUCTION WASNT SUPPOSE TO TRAP
15843 100112  000167  000014                  JMP     2$                      ;REJOIN CODE
15844 100116  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
15845 100120  005726                          TST     (SP)+                   ;CLEAN UP STACK
15846 100122  022704  100056                  CMP     #1$,R4                  ;
15847 100126  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
15848 100130  104003                          ERROR   +3              ;FPP ERROR
15849                                                                 ;PC WAS INCORRECT
15850                                 ;
15851                                 ;COMMON CODE FOR TRAP AND NO TRAP
15852                                 ;VERIFY STATUS
15853 100132  170203                  2$:     STFPS   R3                      ;SAVE FPS
15854 100134  012700  000200                  MOV     #200,R0                 ;SETUP FPS
15855 100140  170100                          LDFPS   R0                      ;FPS=200
15856 100142  174011                          STD     AC0,(R1)                ;GET RESULT
15857 100144  016200  000024                  MOV     24(R2),R0                       ;GET EXPECTED STATUS
15858 100150  020003                          CMP     R0,R3                   ;VERIFY STATUS
15859 100152  001401                          BEQ     3$                      ;BRANCH IF GOOD
15860 100154  104003                          ERROR   +3              ;FPP ERROR
15861                                                                 ;BAD FPS
15862 100156  010204                  3$:     MOV     R2,R4                           ;POINT TO EXPECTED DATA
15863 100160  062704  000012                  ADD     #12,R4
15864 100164  004767  035224          4$:     JSR     R7,DATVER                               ;VERIFY DATA
15865 100170  005767  102724                  TST     COUNT
15866 100174  001401                          BEQ     5$                      ;BRANCH IF GOOD
15867 100176  104003                          ERROR   +3              ;FPP ERROR
15868                                                                 ;BAD AC0
15869 100200  005737  003030          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
15870 100204  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
15871 100206  000162  000026                  JMP     26(R2)                  ;RETURN FROM TEST
15872                                 ;VERIFY FEC
15873 100212  012704  003122          7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
15874 100216  170314                          STST    (R4)                    ;SAVE FEC
15875 100220  021462  000026                  CMP     (R4),26(R2)                     ;VERIFY FEC FOR OVERFLOW
15876 100224  001401                          BEQ     8$                      ;BRANCH IF GOOD
15877 100226  104003                          ERROR   +3              ;FPP ERROR
15878                                                                 ;BAD FEC
15879 100230  000162  000030          8$:     JMP     30(R2)          ;RETURN FROM TEST
15880                                 ;
15881 100234                          HOP20:
15882
15885                                 ;
15886                                 ;
15887                                 ;-----------------------------------
15888                                 ;TEST STCDI, STCDL
15889                                 ;
15890                                         ;
15891                                         ;
15892                                         ;
```

FLOATING POINT TESTS

```
15893 100234                                MSCD:
15894
15895                                       ;
15896                                       ;1/AC0=0, INT
15897 100234  005037  003030                        CLR     @#FLAG          ;NO INTERRUPTS
15898 100240  004767  000610                        JSR     R7,SCDSUB               ;DO TEST
15899 100244  000177  000000  000000                .WORD   0177,0,0,0              ;AC0
      100252  000000
15900 100254  000000  177777                        .WORD   0,-1            ;RESULT
15901 100260  007640                                .WORD   7640            ; TEST FPS
15902 100262  007644                                .WORD   7644            ;RESULT FPS
15903                                       ;2/AC0=-0, LONG
15904 100264  005037  003030                        CLR     @#FLAG          ;INTERRUPT
15905 100270  004767  000560                        JSR     R7,SCDSUB               ;DO TEST
15906 100274  100177  177777  177777                .WORD   100177,-1,-1,-1         ;AC0
      100302  177777
15907 100304  000000  000000                        .WORD   0,0             ;RESULT
15908 100310  007700                                .WORD   7700            ; TEST FPS
15909 100312  007704                                .WORD   7704            ;RESULT FPS
15910                                       ;3/EXP=100, LONG
15911 100314  005037  003030                        CLR     @#FLAG                  ;NO INTERRUPT
15912 100320  004767  000530                        JSR     R7,SCDSUB               ;DO TEST
15913 100324  020000  000000  000000                .WORD   20000,0,0,0             ;AC0
      100332  000000
15914 100334  000000  000000                        .WORD   0,0             ;RESULT
15915 100340  000300                                .WORD   300             ; TEST FPS
15916 100342  000304                                .WORD   304             ;RESULT FPS
15917                                       ;4/EXP=200, BAISED 0, INT, ROUND
15918 100344  005037  003030                        CLR     @#FLAG          ;INTERRUPT
15919 100350  004767  000500                        JSR     R7,SCDSUB               ;DO TEST
15920 100354  140177  177777  000001                .WORD   140177,177777,1,1               ;AC0
      100362  000001
15921 100364  000000  000000                        .WORD   0,0             ;RESULT
15922 100370  007700                                .WORD   7700            ; TEST FPS
15923 100372  007704                                .WORD   7704            ;RESULT FPS
15924                                       ;5/LONG
15925 100374  005037  003030                        CLR     @#FLAG          ;INTERRUPT
15926 100400  004767  000450                        JSR     R7,SCDSUB               ;DO TEST
15927 100404  047667  075757  157737                .WORD   47667,75757,157737,167773               ;AC0
      100412  167773
15928 100414  055675  173757                        .WORD   55675,173757            ;RESULT
15929 100420  007717                                .WORD   7717            ; TEST FPS
15930 100422  007700                                .WORD   7700            ;RESULT FPS
15931                                       ;6/LONG, EXP=2**32
15932 100424  005037  003030                        CLR     @#FLAG                  ;NO INTERRUPT
15933 100430  004767  000420                        JSR     R7,SCDSUB               ;DO TEST
15934 100434  046400  000000  000000                .WORD   46400,0,0,0             ;AC0
      100442  000000
15935 100444  001000  000000                        .WORD   1000,0          ;RESULT
15936 100450  007700                                .WORD   7700            ; TEST FPS
15937 100452  007700                                .WORD   7700            ;RESULT FPS
15938                                       ;7/LONG, EXP>2**32
15939 100454  012737  000001  003030                MOV     #1,@#FLAG               ;INTERRUPT
15940 100462  004767  000366                        JSR     R7,SCDSUB               ;DO TEST
15941 100466  077607  000000  000000                .WORD   77607,0,0,0             ;AC0
      100474  000000
15942 100476  000000  000000                        .WORD   0,0             ;RESULT
```

FLOATING POINT TESTS

```
15943 100502  007700                          .WORD   7700            ; TEST FPS
15944 100504  107705                          .WORD   107705          ;RESULT FPS
15945                               ;8/INT, EXP=2**15
15946 100506  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
15947 100512  004767  000336                  JSR     R7,SCDSUB               ;DO TEST
15948 100516  043200  000000  000000          .WORD   43200,0,0,0             ;AC0
      100524  000000
15949 100526  010000  177777                  .WORD   10000,-1                ;RESULT
15950 100532  007600                          .WORD   7600            ; TEST FPS
15951 100534  007600                          .WORD   7600            ;RESULT FPS
15952                               ;9/INT, EXP>2**15
15953 100536  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
15954 100544  004767  000304                  JSR     R7,SCDSUB               ;DO TEST
15955 100550  077777  177777  177777          .WORD   77777,-1,-1,-1          ;AC0
      100556  177777
15956 100560  000000  177777                  .WORD   0,-1            ;RESULT
15957 100564  007600                          .WORD   7600            ; TEST FPS
15958 100566  107605                          .WORD   107605          ;RESULT FPS
15959                               ;10/INT, EXP>2**15, FID
15960 100570  012737  000000  003030          MOV     #0,@#FLAG               ;NO INTERRUPT
15961 100576  004767  000252                  JSR     R7,SCDSUB               ;DO TEST
15962 100602  043300  000000  000000          .WORD   43300,0,0,0             ;AC0
      100610  000000
15963 100612  000000  014000                  .WORD   0,14000         ;RESULT
15964 100616  047700                          .WORD   47700           ; TEST FPS
15965 100620  047700                          .WORD   47700           ;RESULT FPS
15966                               ;11/INT, EXP>2**15, FIC=0
15967 100622  012737  000000  003030          MOV     #0,@#FLAG               ;NO INTERRUPT
15968 100630  004767  000220                  JSR     R7,SCDSUB               ;DO TEST
15969 100634  143300  177777  177777          .WORD   143300,-1,-1,-2         ;AC0
      100642  177777
15970 100644  177777  163741                  .WORD   -1,163741               ;RESULT
15971 100650  007300                          .WORD   7300            ; TEST FPS
15972 100652  007310                          .WORD   7310            ;RESULT FPS
15973                               ;12/LONG, EXP>2**32, FID
15974 100654  012737  000002  003030          MOV     #2,@#FLAG               ;INTERRUPT
15975 100662  004767  000166                  JSR     R7,SCDSUB               ;DO TEST
15976 100666  050100  000000  000000          .WORD   50100,0,0,0             ;AC0
      100674  000000
15977 100676  000000  000000                  .WORD   0,0             ;RESULT
15978 100702  047700                          .WORD   47700           ; TEST FPS
15979 100704  147705                          .WORD   147705          ;RESULT FPS
15980                               ;13/LONG, EXP>2**32, FIC=0
15981 100706  012737  000000  003030          MOV     #0,@#FLAG               ;NO INTERRUPT
15982 100714  004767  000134                  JSR     R7,SCDSUB               ;DO TEST
15983 100720  050377  177777  177777          .WORD   50377,-1,-1,-1          ;AC0
      100726  177777
15984 100730  000000  000000                  .WORD   0,0             ;RESULT
15985 100734  007300                          .WORD   7300            ; TEST FPS
15986 100736  007305                          .WORD   7305            ;RESULT FPS
15987                               ;14/LONG, EXP<0
15988 100740  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
15989 100744  004767  000104                  JSR     R7,SCDSUB               ;DO TEST
15990 100750  100200  177777  177777          .WORD   100200,-1,-1,-1         ;AC0
      100756  177777
15991 100760  000000  000000                  .WORD   0,0             ;RESULT
15992 100764  007757                          .WORD   7757            ; TEST FPS
```

```
100766  007744                          .WORD   7744            ;RESULT FPS
                                ;15/INT, EXP<0
100770  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
100774  004767  000054                  JSR     R7,SCDSUB               ;DO TEST
101000  037700  177777  177777          .WORD   37700,-1,-1,-2          ;ACO
101006  177776
101010  000000  177777                  .WORD   0,-1            ;RESULT
101014  007600                          .WORD   7600            ; TEST FPS
101016  007604                          .WORD   7604            ;RESULT FPS
                                ;16/INT, EXP=10
101020  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
101024  004767  000024                  JSR     R7,SCDSUB               ;DO TEST
101030  004377  177777  177777          .WORD   4377,-1,-1,-1           ;ACO
101036  177777
101040  000000  177777                  .WORD   0,-1            ;RESULT
101044  007600                          .WORD   7600            ; TEST FPS
101046  007604                          .WORD   7604            ;RESULT FPS
                                ;

101050  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
                                ;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
                                ;
101054  012602                  SCDSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
101056  012737  101144  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
101064  012701  003144                  MOV     #RECDST+2,R1            ;POINT TO RESULT AREA
101070  012711  177777                  MOV     #-1,(R1)                ;PRELOAD RECEIVE DATA BUFFER
101074  012741  177777                  MOV     #-1,-(R1)               ;
101100  012700  000200                  MOV     #200,R0                 ;SET FPS TO DOUBLE
101104  170100                          LDFPS   R0                      ;
101106  010204                          MOV     R2,R4                   ;POINT TO ACO DATA
101110  172414                          LDD     (R4),ACO                        ;LOAD ACO
101112  016200  000014                  MOV     14(R2),R0               ;GET TEST FPS
101116  170100                          LDFPS   R0                      ;LOAD TEST FPS
                                ;
101120  175411                  40$:    STCDI   ACO,(R1)                ;*TEST INSTRUCTION(ACCORDING TO MODE)
101122  170327                  1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA TRAP.
101124  000000                          .WORD   0                       ;STORE STATUS HERE.
                                ;
                                ;
                                ;INSTRUCTION DIDNT TRAP
101126  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
101134  001420                          BEQ     2$                      ;BRANCH IF GOOD
101136  104003                          ERROR   +3              ;FPP ERROR
                                                                ;INSTRUCTION SHOULD HAVE TRAPPED
101140  000167  000032                  JMP     2$                      ;REJOIN CODE
```

```
16048                                   ;
16049                                   ;INSTRUCTION TRAPPED
16050 101144  032737  000001  003030  50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
16051 101152  001003                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
16052 101154  104003                          ERROR   .3              ;FPP ERROR
16053                                         ;INSTRUCTION WASNT SUPPOSE TO TRAP
16054 101156  000167  000014                  JMP     2$                      ;REJOIN CODE
16055 101162  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
16056 101164  005726                          TST     (SP)+                   ;CLEAN UP STACK
16057 101166  022704  101122                  CMP     #1$,R4                  ;
16058 101172  001401                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
16059 101174  104003                          ERROR   .3              ;FPP ERROR
16060                                         ;PC WAS INCORRECT
16061                                   ;
16062                                   ;COMMON CODE FOR TRAP AND NO TRAP
16063                                   ;VERIFY STATUS
16064 101176  170203                  2$:     STFPS   R3                      ;SAVE FPS
16065 101200  016200  000016                  MOV     16(R2),R0                       ;GET EXPECTED STATUS
16066 101204  020003                          CMP     R0,R3                   ;VERIFY STATUS
16067 101206  001401                          BEQ     3$                      ;BRANCH IF GOOD
16068 101210  104003                          ERROR   .3              ;FPP ERROR
16069                                         ;BAD FPS
16070 101212  010204                  3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
16071 101214  062704  000010                  ADD     #10,R4
16072 101220  004767  034152          4$:     JSR     R7,DATVFR                       ;VERIFY DATA
16073 101224  005767  101670                  TST     COUNT
16074 101230  001401                          BEQ     5$                      ;BRANCH IF GOOD
16075 101232  104003                          ERROR   .3              ;FPP ERROR
16076                                         ;BAD ACO
16077 101234  005737  003030          5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
16078 101240  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
16079 101242  000162  000020                  JMP     20(R2)                  ;RETURN FROM TEST
16080                                 ;VERIFY FEC
16081 101246  012704  003122          7$:     MOV     @#RECFEC,R4             ;POINT TO FEC AREA
16082 101252  170314                          STST    (R4)                    ;SAVE FEC
16083 101254  021427  000006                  CMP     (R4),#6                 ;VERIFY FEC FOR OVERFLOW
16084 101260  001401                          BEQ     8$                      ;BRANCH IF GOOD
16085 101262  104003                          ERROR   .3              ;FPP ERROR
16086                                                                         ;BAD FEC
16087 101264  000162  000020          8$:     JMP     20(R2)          ;RETURN FROM TEST
16088                                 ;
16089 101270                          MOP21:
16090                                 ;
16093                                 ;
16094                                 ;
16095                                 ;-----------------------------------
16096                                 ;TEST STCFI, STCFL
16097                                 ;
16098                                         ;
16099                                         ;
16100                                         ;
16101 101270                          MSCF:
16102
16103                                 ;
16104                                 ;1/LONG EXP -30
16105 101270  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
16106 101274  004767  177554                  JSR     R7,SCDSUB               ;DO TEST
```

FLOATING POINT TESTS

```
16107 101300  044541  052525  177777          .WORD   44541,52525,-1,-1                ;AC0
      101306  177777
16108 101310  000003  102525                   .WORD   3,102525                    ;RESULT
16109 101314  007517                           .WORD   7517                ; TEST FPS
16110 101316  007500                           .WORD   7500                ;RESULT FPS
16111                               ;2/INT, EXP<0
16112 101320  005037  003030                   CLR     @#FLAG              ;NO INTERRUPTS
16113 101324  004767  177524                   JSR     R7,SCDSUB                   ;DO TEST
16114 101330  002300  177777  177777          .WORD   2300,-1,-1,-1               ;AC0
      101336  177777
16115 101340  000000  177777                   .WORD   0,-1                ;RESULT
16116 101344  007400                           .WORD   7400                ; TEST FPS
16117 101346  007404                           .WORD   7404                ;RESULT FPS
16118                               ;3/LONG, EXP
16119 101350  012737  000001  003030           MOV     #1,@#FLAG                   ;INTERRUPT
16120 101356  004767  177472                   JSR     R7,SCDSUB                   ;DO TEST
16121 101362  070000  177777  177777          .WORD   70000,-1,-1,-1              ;AC0
      101370  177777
16122 101372  000000  000000                   .WORD   0,0                 ;RESULT
16123 101376  007540                           .WORD   7540                ; TEST FPS
16124 101400  107545                           .WORD   107545              ;RESULT FPS
16125                               ;4/INT,EXP=5, FIC=0, FID=1
16126 101402  005037  003030                   CLR     @#FLAG              ;NO INTERRUPTS
16127 101406  004767  177442                   JSR     R7,SCDSUB                   ;DO TEST
16128 101412  052000  000000  177777          .WORD   52000,0,-1,-1               ;AC0
      101420  177777
16129 101422  000000  177777                   .WORD   0,-1                ;RESULT
16130 101426  047000                           .WORD   47000               ; TEST FPS
16131 101430  047005                           .WORD   47005               ;RESULT FPS
16132                               ;
16133
16134
16137                               ;
16138                               ;
16139                               ;---------------------------------
16140                               ;TEST STEXP
16141                               ;
16142                                       ;
16143                                       ;
16144                                       ;
16145 101432                        MSXP:
16146
16147                               ;
16148                               ;1/EXP=100
16149 101432  004767  000154                   JSR     R7,SXPSUB                   ;DO TEST
16150 101436  020000  000000  000000           .WORD   20000,0,0,0                 ;AC0
      101444  000000
16151 101446  177700                           .WORD   -100                ;RESULT
16152 101450  007740                           .WORD   7740                ; TEST FPS
16153 101452  007750                           .WORD   7750                ;RESULT FPS
16154                               ;2/EXP=201 FLOAT, NEG
16155 101454  004767  000132                   JSR     R7,SXPSUB                   ;DO TEST
16156 101460  140377  177777  177777          .WORD   140377,-1,-1,0              ;AC0
      101466  000000
16157 101470  000001                           .WORD   1                   ;RESULT
16158 101472  007500                           .WORD   7500                ; TEST FPS
16159 101474  007500                           .WORD   7500                ;RESULT FPS
```

FLOATING POINT TESTS

```
16160                                   ;3/EXP=-177
16161 101476  004767  000110                      JSR     R7,SXPSUB               ;DO TEST
16162 101502  000177  177777  177777              .WORD   177,-1,-1,-1            ;AC0
      101510  177777
16163 101512  177600                              .WORD   177600          ;RESULT
16164 101514  007700                              .WORD   7700            ; TEST FPS
16165 101516  007710                              .WORD   7710            ;RESULT FPS
16166                                   ;4/EXP=-100
16167 101520  004767  000066                      JSR     R7,SXPSUB               ;DO TEST
16168 101524  020000  000000  177777              .WORD   20000,0,-1,-1           ;AC0
      101532  177777
16169 101534  177700                              .WORD   -100            ;RESULT
16170 101536  040200                              .WORD   40200           ; TEST FPS
16171 101540  040210                              .WORD   40210           ;RESULT FPS
16172                                   ;5/EXP=200
16173 101542  004767  000044                      JSR     R7,SXPSUB               ;DO TEST
16174 101546  040000  000000  000000              .WORD   40000,0,0,0             ;AC0
      101554  000000
16175 101556  000000                              .WORD   0               ;RESULT
16176 101560  007700                              .WORD   7700            ; TEST FPS
16177 101562  007704                              .WORD   7704    ;RESULT FPS
16178                                   ;6/EXP=0
16179 101564  004767  000022                      JSR     R7,SXPSUB               ;DO TEST
16180 101570  000177  177777  177777              .WORD   177,-1,-1,-1            ;AC0
      101576  177777
16181 101600  177600                              .WORD   177600          ;RESULT
16182 101602  000000                              .WORD   0               ; TEST FPS
16183 101604  000010                              .WORD   10              ;RESULT FPS
16184                                   ;
16185
16186 101606  000167  000104                      JMP     HOP22           ;GET OVER SUBROUTINE
16187                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16188                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16189                                   ;STEXP
16190                                   ;               AC0
16191                                   ;               EXPONENT RESULT
16192                                   ;               FPS BEFORE EXECUTION
16193                                   ;               FPS AFTER EXECUTION
16194                                   ;
16195                                   ;NO TRAPS CAN OCCUR
16196                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16197                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16198                                   ;
16199 101612  012602                    SXPSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
16200 101614  012737  101704  000244            MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
16201 101622  012701  003142                    MOV     #RECDST,R1              ;POINT TO RESULT AREA
16202 101626  012700  000200                    MOV     #200,R0                 ;SET FPS TO DOUBLE
16203 101632  170100                            LDFPS   R0                      ;
16204 101634  010204                            MOV     R2,R4                   ;POINT TO AC0 DATA
16205 101636  172414                            LDD     (R4),AC0                        ;LOAD AC0
16206 101640  016200  000012                    MOV     12(R2),R0               ;GET TEST FPS
16207 101644  170100                            LDFPS   R0                      ;LOAD TEST FPS
16208                                   ;
16209 101646  175011                    40$:    STEXP   AC0,(R1)                ;*TEST INSTRUCTION(ACCORDING TO MODE)
16210                                   ;
16211                                   ;VERIFY STATUS
16212 101650  170203                    2$:     STFPS   R3                      ;SAVE FPS
```

FLOATING POINT TESTS

```
16213 101652  016200  000014                  MOV     14(R2),R0               ;GET EXPECTED STATUS
16214 101656  020003                          CMP     R0,R3                   ;VERIFY STATUS
16215 101660  001401                          BEQ     3$                      ;BRANCH IF GOOD
16216 101662  104003                          ERROR   .+3                  ;FPP ERROR
16217                                                                      ;BAD FPS
16218 101664  016204  000010          3$:     MOV     10(R2),R4               ;POINT TO EXPECTED EXPONENT
16219 101670  020437  003142                  CMP     R4,@#RECDST             ;VERIFY EXPONENT
16220 101674  001401                          BEQ     5$                      ;BRANCH IF GOOD
16221 101676  104003                          ERROR   .+3                  ;FPP ERROR
16222                                                                      ;BAD ACO
16223 101700  000162  000016          5$:     JMP     16(R2)                  ;RETURN FROM TEST
16224                                         ;
16225                                         ;INSTRUCTION TRAPPED
16226 101704  012600                  50$:    MOV     (SP)+,R0                ;SAVE PC
16227 101706  012605                          MOV     (SP)+,R5                ;SAVE OLD PS
16228 101710  104003                          ERROR   .+3                  ;FPP ERROR
16229                                                                      ;WILD TRAP DURING STEXP
16230 101712  000167  177762                  JMP     5$                      ;REJOIN CODE
16231                                         ;
16232
16233 101716                          MOP22:
16236
16237
16238                                 .ENABL AMA
16239                                 .SBTTL  TEST  - CHECK REGISTER ACCESS
16240                                 ;CHECK REGISTER ACCESS - THIS TEST WILL VERIFY EACH OF THE THREE
16241                                 ;CACHE MEMORY SYSTEM REGISTERS CAN BE ACCESSED WITHOUT A TRAP TO
16242                                 ;LOCATION 4(NON-EXISTANT ADDRESS TRAP) OCCURRING. THE REGISTERS TO
16243                                 ;BE TESTED ARE:
16244                                 ;       CACHE CONTROL           17777746
16245                                 ;       MEMORY SYSTEM ERROR     17777744
16246                                 ;       HIT/MISS                17777752
16247                                 ;EACH REGISTER WILL BE ACCESSED WITH A WRITE CYCLE.
16248                                 ;
16249                                 ;BGNTST
16250                                 ;SAVE CONTENT OF LOCATION 4
16251                                 ;LET LOCATION 4 POINT TO ERROR ROUTINE
16252                                 ;INITIALIZE R TO TOP OF ADDRESS TABLE
16253                                 ;DO UNTIL FOR ALL REGISTERS
16254                                 ;.      TEST REGISTER UNDER TEST
16255                                 ;.      IF TIMEOUT DID HAPPEN
16256                                 ;.      .       ERROR
16257                                 ;.      ENDIF
16258                                 ;ENDDO
16259                                 ;RESTORE CONTENTS OF LOCATION 4
16260                                 ;EXIT   TST
16261                                 ;
16262                                 ;ADDRESS TABLE: 17777746
16263                                 ;               17777744
16264                                 ;               17777752
16265                                 ;
16266                                 ;ENDTST
16267                                 ;ERROR ROUTINE: SET TIMEOUT FLAG
16268                                 ;               RETURN
16269
16270                                 ;;*****************************************************************
      101716  000004                  TST2:   SCOPE
```

TEST - CHECK REGISTER ACCESS

```
16271 101720  000240                        NOP
16272 101722  005737  003032                TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16273 101726  001002                        BNE     99$                     ; NOT YET
16274
16275 101730  000240                        NOP                             ; DEBUG AID
16276 101732  000453                        BR      TST3            ;;GO TO NEXT TEST
16277 101734  000240                99$:    NOP
16278 101736  012737  135056  000004        MOV     #TOUT,  @#4
16279 101744  013737  000004  003012        MOV     @#4,    SLOC00          ;SAVE CONTENTS OF VECTOR 4
16280 101752  012737  102056  000004        MOV     #ERROUT,@#4             ;LET VECTOR 4 POINT TO ERROR ROUTINE
16281 101760  005001                        CLR     R1                      ;CLEAR TIMEOUT FLAG
16282 101762  005737  177746          1$:   TST     CCR                     ;READ REGISTER UNDER TEST
16283 101766  005701                        TST     R1                      ;TIMEOUT?
16284 101770  001404                        BEQ     2$
16285 101772  012737  177746  001122        MOV     #CCR,$BDADR             ;STORE LOCATION
16286 102000  104130                        ERROR   +130                    ;TIMEOUT ACCESING CCR
16287 102002  005001                2$:     CLR     R1                      ;CLEAR TIMOUT FLAG
16288 102004  005737  177744                TST     MSER                    ;READ MSER
16289 102010  005701                        TST     R1                      ;TIMEOUT ?
16290 102012  001404                        BEQ     3$
16291 102014  012737  177744  001122        MOV     #MSER,$BDADR            ;STORE LOCATION
16292 102022  104130                        ERROR   +130                    ;TIMEOUT ACCESSING MSER
16293 102024  005001                3$:     CLR     R1                      ;CLEAR TIMEOUT FLAG
16294 102026  005737  177752                TST     HITMIS                  ;ACCESS HIT/MISS
16295 102032  005701                        TST     R1                      ;TIMEOUT?
16296 102034  001404                        BEQ     4$
16297 102036  012737  177752  001122        MOV     #HITMIS,$BDADR          ;STORE LOCATION
16298 102044  104130                        ERROR   +130                    ;TIMEOUT ACCESSING HIT/MISS
16299 102046  013737  003012  000004  4$:   MOV     SLOC00, @#4             ;RESTORE VECTOR 4
16300 102054  000402                        BR      TST3            ;;GO TO NEXT TEST
16301
16302 102056                        ERROUT:
16303 102056  005201                        INC     R1                              ;FLAG TIMEOUT
16304 102060  000002                        RTI
16305
```

```
                                        .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

                                        ;;*****************************************************************
      102062  000004                    TST3:   SCOPE
      102064  000240                            NOP
      102066  005737  003032                    TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
      102072  001002                            BNE     99$                     ; NOT YET
      102074  000240                            NOP                             ; DEBUG AID
      102076  000431                            BR      TST4            ;;GO TO NEXT TEST
      102100  000240                    99$:    NOP
      102102  012701  000001                    MOV     #1,     R1              ;INITIALIZE GOOD DATA TO #1
      102106  005037  177746                    CLR     CCR                     ;CLEAR CCR
      102112  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
      102120  010137  177746            1$:     MOV     R1,     CCR             ;WRITE GOOD DATA TO CCR
      102124  023701  177746                    CMP     CCR,    R1              ;IF CCR NOT EQUAL GOOD DATA
      102130  001407                            BEQ     3$                      ;THEN
      102132  005737  177746                    TST     CCR                     ;IF CCR EQUAL TO ZERO
      102136  001003                            BNE     2$                      ;THEN
      102140  032701  174400                    BIT     #174400,R1              ;IF GOOD DATA NE TO BIT 15-11 OR BIT 8
      102144  001001                            BNE     3$                      ;THEN
      102146  104004                    2$:     ERROR   +4                      ;ERROR IN READ/WRITE BITS OF CCR
      102150  006101                    3$:     ROL     R1                      ;UPDATE TO NEXT BIT
      102152  103362                            BCC     1$                      ;DO UNTIL ALL BITS TESTED
      102154  052737  001000  177520            BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
```

TEST - FORCE MISS TEST

```
16352                           .SBTTL  TEST  - FORCE MISS TEST
16353                           ;FORCE MISS TEST - THIS TEST WILL VERIFY THAT ALL REFERENCES MADE
16354                           ;WITH EITHER BIT<3> OR BIT<2> OF THE CCR SET CAUSE A CACHE MISS AND
16355                           ;LEAVE THE CACHE ENTRY UNCHANGED. FIRST WRITE A TEST ADDRESS WITH
16356                           ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN
16357                           ;INTO THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH
16358                           ;NEW DATA. NEXT CLEAR BITS<3:2> AND READ THE TEST ADDRESS, THE DATA
16359                           ;SHOULD EQUAL TO PATTERN 1. FINALLY READ THE TEST ADDRESS WITH BIT<3>
16360                           ;SET, THE DATA SHOULD EQUAL PATTERN 2. A LAST WRITE MUST BE DONE WITH
16361                           ;BOTH FORCE BITS CLEARED BECAUSE THE CACHE AND MAIN MEMORY HAVE
16362                           ;DIFFERENT DATA.
16363                           ;
16364                           ;
16365                           ;BGNTST
16366                           ;WRITE TEST ADDRESS WITH PATTERN 1
16367                           ;SET CCR BITS<3:2> = 0,1
16368                           ;WRITE TEST ADDRESS WITH PATTERN 2
16369                           ;CLEAR CCR BIT<3>
16370                           ;READ TEST ADDRESS
16371                           ;SAVE HIT/MISS REGISTER DATA
16372                           ;COMPARE RECEIVED DATA TO PATTERN 1
16373                           ;IF DATA NOT EQUAL THEN
16374                           ;.      IF DATA EQUAL TO PATTERN 2 THEN
16375                           ;.      .       IF HIT THEN
16376                           ;.      .       .       ERROR FORCE MISS WRITES TO CACHE
16377                           ;.      .       ELSE
16378                           ;.      .       .       IF PARITY ERROR INDICATORS EQUAL 0 THEN
16379                           ;.      .       .       .       ERROR FROCE MISS INVALIDATES CACHE
16380                           ;.      .       .       ELSE
16381                           ;.      .       .       .       IF ALL PARITY INDICATORS SET THEN
16382                           ;.      .       .       .       .       ERROR PARITY ABORT AFTER FORCE MISS
16383                           ;.      .       .       .       ENDIF
16384                           ;.      .       .       .       IF TAG PARITY ERROR THEN
16385                           ;.      .       .       .       .       ERROR  TAG PARITY ERROR AFTER FORCE MISS
16386                           ;.      .       .       .       ELSE
16387                           ;.      .       .       .       .       IF B0 AND  B1 ERROR THEN
16388                           ;.      .       .       .       .       .       ERROR DATA PARITY ERROR AFTER FORCE MISS
16389                           ;.      .       .       .       .       ENDIF
16390                           ;.      .       .       .       .       IF B0 PARITY ERROR THEN
16391                           ;.      .       .       .       .       .       ERROR LOW BYTE PARITY ERROR
16392                           ;.      .       .       .       .       ENDIF
16393                           ;.      .       .       .       .       IF B1 PARITY ERROR THEN
16394                           ;.      .       .       .       .       .       ERROR HIGH BYTE PARITY ERROR
16395                           ;.      .       .       .       .       ENDIF
16396                           ;.      .       .       .       ENDIF
16397                           ;.      .       .       ENDIF
16398                           ;.      .       ENDIF
16399                           ;.      ELSE
16400                           ;.      .       ERROR IN DATA PATH
16401                           ;.      ENDIF
16402                           ;ENDIF
16403                           ;SET CCR<3:2> = 1,0
16404                           ;READ TEST ADDRESS
16405                           ;SAVE HIT/MISS REGISTER DATA
16406                           ;COMPARE RECEIVED DATA TO PATTERN 2
16407                           ;IF DATA NOT EQUAL THEN
16408                           ;.      IF RECIEVED DATA EQUAL TO PATTERN 1 THEN
```

```
16409                                   ;.      .       IF HIT THEN
16410                                   ;.      .           ERROR FORCE MISS READS FROM CACHE
16411                                   ;.      .       ELSE
16412                                   ;.      .           ERROR FORCE MISS READS FROM CACHE AND MISS
16413                                   ;.      .       ENDIF
16414                                   ;.      ELSE
16415                                   ;.          ERROR IN DATA PATH
16416                                   ;.      ENDIF
16417                                   ;ENDIF
16418                                   ;CLEAR CCR<3:2>
16419                                   ;WRITE TEST ADDRESS
16420                                   ;ENDTST
16421                                   ;
16422                                   ;;*****************************************************************
     102162  000004                   TST4:   SCOPE
16423 102164  000240                           NOP
16424 102166  005737  003032                   TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16425 102172  001002                           BNE     99$                     ; NOT YET
16426 102174  000240                           NOP                             ; DEBUG AID
16427 102176  000563                           BR      TST5            ;;GO TO NEXT TEST
16428 102200  000240                   99$:    NOP
16429 102202  013737  000114  003012           MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16430 102210  012737  102534  000114           MOV     #FMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
16431 102216  005003                           CLR     R3                      ;CLEAR MSER SAVE LOCATION
16432 102220  005037  003162                   CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16433 102224  012737  000004  177746           MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
16434 102232  012737  177777  003162           MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16435 102240  012737  000200  177746           MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16436 102246  012737  177777  001124           MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
16437 102254  042737  001000  177520           BIC     #1000,@#BCSR            ;DISABLE HALT ON BREAK
16438 102262  013701  003162                   MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
16439 102266  013702  177752                   MOV     @#HITMIS,R2             ;SAVE HIT/MISS DATA
16440 102272  052737  001000  177520           BIS     #1000,@#BCSR            ;ENABLE HALT ON BREAK
16441 102300  005701                           TST     R1                      ;COMPARE RECEIVED DATA TO PATTERN 1
16442 102302  001451                           BEQ     1$                      ;IF DATAS NOT EQUAL THEN
16443 102304  022701  177777                   CMP     #177777,R1              ;IF DATA EQUAL TO PATTERN 2
16444 102310  001045                           BNE     106$                    ;THEN
16445 102312  032702  000002                   BIT     #BIT01, R2              ;IF HIT
16446 102316  001402                           BEQ     100$                    ;THEN
16447 102320  104005                           ERROR   .5                      ;FORCE MISS WRITES TO CACHE
16448 102322  000441                           BR      1$                      ;ELSE
16449 102324  032703  100340           100$:   BIT     #100340,R3              ;IF PARITY INDICATORS EQUAL 0
16450 102330  001002                           BNE     101$                    ;THEN
16451 102332  104006                           ERROR   .6                      ;FORCE MISS WRITE INVALIDATES CACHE
16452 102334  000434                           BR      1$                      ;ELSE
16453 102336  022703  100340           101$:   CMP     #100340,R3              ;IF ALL PARITY INDICATORS SET
16454 102342  001002                           BNE     102$                    ;THEN
16455 102344  104007                           ERROR   .7                      ;PARITY ABORT AFTER FORCE MISS
16456 102346  000427                           BR      1$                      ;ELSE
16457 102350  032703  000040           102$:   BIT     #BIT05, R3              ;IF TAG PARITY ERROR
16458 102354  001402                           BEQ     103$                    ;THEN
16459 102356  104010                           ERROR   .10                     ;TAG PARITY ERROR AFTER FORCE MISS
16460 102360  000422                           BR      1$                      ;ELSE
16461 102362  010304                   103$:   MOV     R3,     R4              ;COPY MSER INTO REG 4
16462 102364  042704  177477                   BIC     #177477,R4              ;MASK FOR B0 AND B1 DATA
16463 102370  022704  000300                   CMP     #300,   R4              ;IF B0 AND B1 DATA
16464 102374  001002                           BNE     104$                    ;THEN
```

TEST - FORCE MISS TEST

```
16465 102376  104011                        ERROR   +11                 ;DATA PARITY ERROR AFTER FORCE MISS
16466 102400  000412                        BR      1$                  ;ELSE
16467 102402  032703  000100          104$: BIT     #100,   R3          ;IF B0 ERROR
16468 102406  001401                        BEQ     105$                ;THEN
16469 102410  104012                        ERROR   +12                 ;LOW BYTE PARITY ERROR AFTER FORCE MISS
16470 102412  032703  000200          105$: BIT     #200,   R3          ;IF B1 ERROR
16471 102416  001403                        BEQ     1$                  ;THEN
16472 102420  104013                        ERROR   +13                 ;HIGH BYTE PARITY ERROR AFTER FORCE MISS
16473 102422  000401                        BR      1$                  ;ENDIF
16474 102424  104014                  106$: ERROR   +14                 ;ERROR DATA PATH
16475 102426  005003                  1$:   CLR     R3                  ;CLEAR MSER SAVE REGISTER
16476 102430  052737  000010  177746        BIS     #BIT03, CCR         ;SET CCR BITS <3:2> = 1,0
16477 102436  042737  001000  177520        BIC     #1000,BCSR          ;DISABLE HALT ON BREAK
16478 102444  013701  003162              MOV     @#TSTLOC,R1         ;READ TEST LOCATION
16479 102450  013702  177752              MOV     @#HITMIS,R2         ;SAVE HIT/MISS REGISTER DATA
16480 102454  052737  001000  177520        BIS     #1000,BCSR          ;ENABLE HALT ON BREAK
16481 102462  020127  177777              CMP     R1,     #177777     ;COMPARE RECEIVED DATA TO PATTERN 2
16482 102466  001412                        BEQ     2$                  ;IF DATAS NOT EQUAL THEN
16483 102470  005701                        TST     R1                  ;IF RECIEVED DATA EQUAL TO PATTERN 1
16484 102472  001007                        BNE     108$                ;THEN
16485 102474  032702  000002              BIT     #BIT01, R2          ;IF HIT
16486 102500  001402                        BEQ     107$                ;THEN
16487 102502  104015                        ERROR   +15                 ;FORCE MISS READS FROM CACHE
16488 102504  000403                        BR      2$                  ;ELSE
16489 102506  104016                  107$: ERROR   +16                 ;FORCE MISS READS FROM CACHE AND MISS
16490 102510  000401                        BR      2$                  ;ENDIF
16491 102512  104014                  108$: ERROR   +14                 ;ERROR IN DATA PATH
16492 102514  013737  003012  000114  2$:   MOV     SLOC00, @#114       ;RESTORE VECTOR 114
16493 102522  005037  177746              CLR     CCR                 ;CLEAR CCR BITS<3:2>
16494 102526  005037  003162              CLR     @#TSTLOC            ;WRITE TEST ADDRESS
16495 102532  000405                        BR      TST5            ;;GO TO NEXT TEST
16496
16497
16498 102534  011637  001122        FMPARR: MOV     (SP),@#BDADR        ;SAVE ADDRESS THAT CAUSED ABORT
16499 102540  013703  177744              MOV     MSER,   R3          ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
16500 102544  000002                        RTI                         ;REGISTER AND RETURN
16501
16502
16503                                 .SBTTL  TEST  - HIT/MISS REGISTER TEST PART 1
16504                                 ;HIT/MISS REGISTER TEST PART 1 - THIS TEST WILL VERIFY THAT THE HIT/MISS
16505                                 ;REGISTER CORRECTLY LOGS HITS AND MISSES.FIRST WRITE A TEST ADDRESS WITH
16506                                 ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN INTO
16507                                 ;THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH NEW DATA.
16508                                 ;NEXT CLEAR CCR BITS<3:2> AND READ THE TEST ADDRESS. THE HIT/MISS REGISTER
16509                                 ;SHOULD LOGGED A HIT. FINALLY READ THE TEST ADDRESS PLUS 20000(8) THE
16510                                 ;HIT/MISS REGISTER SHOULD HAVE LOGGED A MISS.
16511                                 ;
16512                                 ;
16513                                 ;BGNTST
16514                                 ;WRITE TEST ADDRESS WITH PATTERN 1
16515                                 ;SET CCR BITS<3:2> = 0,1
16516                                 ;WRITE TEST ADDRESS WITH PATTERN 2
16517                                 ;CLEAR CCR BIT<3>
16518                                 ;READ TEST ADDRESS
16519                                 ;IF HIT/MISS REGISTER BIT 1 NOT SET THEN
16520                                 ;.      ERROR IN RECORDING HITS IN HIT/MISS
16521                                 ;ENDIF
```

TEST - HIT/MISS REGISTER TEST PART 1

```
16522                                   ;READ TEST ADDRESS + 20000(8)
16523                                   ;IF HIT/MISS REGISTER BIT 1 SET THEN
16524                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
16525                                   ;ENDIF
16526                                   ;ENDTST
16527                                   ;
16528                                   ;;*****************************************************************
     102546  000004                     TST5:   SCOPE
16529
16530 102550  000240                             NOP
16531 102552  005737  003032                     TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16532 102556  001002                             BNE     99$                     ; NOT YET
16533 102560  000240                             NOP                             ; DEBUG AID
16534 102562  000463                             BR      TST6            ;;GO TO NEXT TEST
16535 102564  000240                     99$:    NOP
16536 102566  013737  000114  003012             MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16537 102574  012737  102724  000114             MOV     #HMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
16538 102602  005037  003162                     CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16539 102606  012737  000004  177746             MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
16540 102614  012737  177777  003162             MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16541 102622  012737  000200  177746             MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16542 102630  042737  001000  177520             BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16543 102636  013701  003162                     MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
16544 102642  013737  177752  111464             MOV     HITMIS,RECDAT           ;STORE REGISTER
16545 102650  032737  000004  111464             BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 NOT SET
16546 102656  001001                             BNE     1$                      ;THEN
16547 102660  104045                             ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
16548 102662  013701  023162             1$:     MOV     @#TSTLOC+8192.,R1       ;READ TEST LOCATION + 20000(8)
16549 102666  013737  177752  111464             MOV     HITMIS,RECDAT           ;STORE REGISTER
16550 102674  032737  000004  111464             BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
16551 102702  001401                             BEQ     2$                      ;THEN
16552 102704  104045                             ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
16553 102706  013737  003012  000114     2$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
16554 102714  052737  001000  177520             BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16555 102722  000403                             BR      TST6            ;;GO TO NEXT TEST
16556
16557 102724  013703  177744             HMPARR: MOV     MSER,   R3              ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
16558 102730  000002                             RTI                             ;REGISTER AND RETURN
16559
```

```
16561                                   .SBTTL  TEST  - HIT/MISS REGISTER TEST
16562                                   ;HIT/MISS REGISTER TEST - THIS TEST WILL VERIFY THAT THE HIT/MISS
16563                                   ;REGISTER CORRECTLY LOGS CACHE HITS AND MISSES. IT WILL ALSO VERIFY
16564                                   ;THAT EACH BIT OF THE REGISTER IS UNIQUE. THIS WILL BE DONE BY
16565                                   ;FLOATING A ZERO THROUGH A FIELD OF ONES. THE ROTATING WILL BE DONE
16566                                   ;BY EXECUTING A TST @#MM (WHERE MM IS THE ADDRESS OF THE HIT/MISS
16567                                   ;REGISTER) AT SUCCESSIVE POSITIONS IN A SET OF EIGHT NOPS. THE READ
16568                                   ;OF THE I/O PAGE ADDRESS OF THE HIT/MISS REGISTER SHOULD CAUSE A
16569                                   ;MISS TO BE RECORDED.
16570                                   ;
16571                                   ;BGNTST
16572                                   ;INITIALIZE LOOP INDICATOR
16573                                   ;INITIALIZE EXPECTED DATA
16574                                   ;DO UNTIL LOOP INDICATOR = SEVEN
16575                                   ;.      PUT BACKGROUND DATA INTO EXECUTION BUFFER
16576                                   ;.      PUT TST INSTRUCTION INTO TEST LOCATION
16577                                   ;.      JUMP TO EXECUTE BUFFER
16578                                   ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
16579                                   ;.      .       ERROR IN RECORDING HITS IN HIT/MISS
16580                                   ;.      ENDIF
16581                                   ;.      INCREMENT LOOP INDICATOR
16582                                   ;.      UPDATE EXPECTED DATA
16583                                   ;ENDDO
16584                                   ;EXIT   TST
16585                                   ;
16586                                   ;BACKGROUND DATA:NOP
16587                                   ;                NOP
16588                                   ;                NOP
16589                                   ;                NOP
16590                                   ;                NOP
16591                                   ;                NOP
16592                                   ;                NOP
16593                                   ;                NOP
16594                                   ;                MOV     @#MM,R2
16595                                   ;                RTS     PC
16596                                   ;ENDTST
16597                                   ;;*****************************************************************************
    102732  000004                    TST6:   SCOPE
16598 102734  000240                            NOP
16599 102736  005737  003032                    TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16600 102742  001002                            BNE     99$                     ; NOT YET
16601 102744  000240                            NOP                             ; DEBUG AID
16602 102746  000511                            BR      TST7            ;;GO TO NEXT TEST
16603 102750  000240                    99$:    NOP
16604 102752  005037  003154                    CLR     LOOPIN                  ;INITIALIZE LOOP INDICATOR
16605 102756  013737  023204  001162            MOV     @#TSTLOC+20022,$TMP1    ;SAVE FOR LATER
16606 102764  012703  003164                    MOV     #TSTLOC+2,R3            ;GET ADDRESS OF EXECUTION BUFFER
16607 102770  012704  103154                    MOV     #EXPTBL,R4              ;GET ADDRESS OF EXPECTED DATA TABLE
16608 102774  012705  177752                    MOV     #HITMIS,R5              ;PUT ADDRESS OF HIT/MISS REGISTER IN GPR
16609 103000  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16610 103006  023727  003154  000007  1$:       CMP     LOOPIN, #7              ;DO UNTIL LOOP INDICATOR EQUALS 7
16611 103014  001440                            BEQ     ENDMRT                  ;THEN EXIT
16612 103016  012702  000013                    MOV     #13,    R2
16613 103022  012700  103126                    MOV     #BACDAT,R0              ;PUT BACKGROUND DATA INTO
16614 103026  012701  003162                    MOV     #TSTLOC,R1              ;EXECUTION BUFFER
16615 103032  012021                    2$:     MOV     (R0)+,  (R1)+
16616 103034  077202                            SOB     R2,     2$              ;LOOP FOR TEN WORDS
```

TEST - HIT/MISS REGISTER TEST

```
16617 103036  023727  003154  000006          CMP     LOOPIN, #6              ;IS THIS LAST LOOP
16618 103044  001004                          BNE     3$                      ;IF YES THEN
16619 103046  013737  001162  023204          MOV     $TMP1,@#TSTLOC+20022    ;INVALIDATE FETCH OF "RECDAT"
16620 103054  000404                          BR      4$                      ;ELSE
16621 103056  012723  005737          3$:     MOV     #5737,  (R3)+           ;MOVE "TST" INSTRUCTION TO BUFFER
16622 103062  012713  177752                  MOV     #HITMIS,(R3)            ;MOVE HIT.MISS REG. ADD. TO BUFFER
16623 103066  004737  003162          4$:     JSR     PC,     TSTLOC          ;GO EXECUTE TEST CODE
16624 103072  021437  111464                  CMP     (R4),   RECDAT          ;IF EXPECTED DATA NOT EQUAL TO
16625 103076  001403                          BEQ     5$                      ;RECEIVED DATA THEN
16626 103100  011437  001124                  MOV     (R4),$GDDAT             ;SAVE EXPECTED PATTERN
16627 103104  104017                          ERROR   +17                     ;ERROR IN RECORDING HITS IN HIT/MISS
16628 103106  005724          5$:     TST     (R4)+                           ;INCREMENT POINTER TO EXPECTED PATTERN
16629 103110  005237  003154                  INC     LOOPIN                  ;INCREMENT LOOP COUNTER
16630 103114  000734                          BR      1$
16631 103116                  ENDHRT:
16632 103116  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16633 103124  000422                          BR      TST7            ;;GO TO NEXT TEST
16634
16635 103126  000240          BACDAT: .WORD   NOP
16636 103130  000240                  .WORD   NOP
16637 103132  000240                  .WORD   NOP
16638 103134  000240                  .WORD   NOP
16639 103136  000240                  .WORD   NOP
16640 103140  000240                  .WORD   NOP
16641 103142  000240                  .WORD   NOP
16642 103144  000240                  .WORD   NOP
16643 103146  011537  111464                  MOV     (R5),RECDAT             ;SAVE CONTENTS OF HIT/MISS REGISTER
16644 103152  000207                          RTS     PC
16645
16646 103154  000077          EXPTBL: .WORD   77
16647 103156  000037                  .WORD   37
16648 103160  000057                  .WORD   57
16649 103162  000067                  .WORD   67
16650 103164  000073                  .WORD   73
16651 103166  000075                  .WORD   75
16652 103170  000076                  .WORD   76
16653
```

```
16655                                   .SBTTL  TEST  - BYTE ALLOCATION TEST
16656                                   ;BYTE ALLOCATION TEST - THIS TEST WILL VERIFY THAT THE CACHE SYSTEM CORRECTLY
16657                                   ;HANDLES BYTE ACCESSES. THE TEST WILL FIRST CHECK THAT A WRITE ACCESS WHICH
16658                                   ;IS A MISS DOES NOT UPDATE THE CACHE. THIS WILL BE DONE BY FIRST ASSURRING
16659                                   ;A MISS AT A TEST LOCATION. THEN A WRITE BYTE ACCESS WILL BE DONE. FINALLY
16660                                   ;THE LOCATION WILL BE READ. THE WRITE BYTE SHOULD NOT HAVE ALLOCATED THE
16661                                   ;ADDRESS. NEXT THE TEST WILL VERIFY THAT IF A WRITE BYTE ACCESS IS A HIT
16662                                   ;THAT THE CACHE LOCATION IS UPDATED. THE TEST WILL FIRST WRITE A TEST LOCATION
16663                                   ;WITH A PATTERN TO ALLOCATE THE ADDRESS. THEN A SECOND PATTERN WILL BE WRITTEN
16664                                   ;TO THE HIGH BYTE OF THE TEST LOCATION. THE TEST LOCATION WILL THEN BE READ.
16665                                   ;IF THE HIGH BYTE OF THE TEST LOCATION IS EQUAL TO THE SECOND PATTERN THEN THE
16666                                   ;LOCATION WAS UPDATED. THE SECOND TEST WILL BE REPEATED TO TEST LOW BYTE
16667                                   ;ACCESSES.
16668                                   ;
16669                                   ;BGNTST
16670                                   ;SAVE VECTOR 114
16671                                   ;LET VECTOR 114 POINT TO BYTE PARITY ROUTINE
16672                                   ;SET PARITY ABORT BIT IN CACHE CONTROL REGISTER
16673                                   ;READ TEST LOCATION + 4K
16674                                   ;WRITE LOW BYTE TO TEST ADDRESS
16675                                   ;READ LOW BYTE OF TEST ADDRESS
16676                                   ;IF HIT/MISS REGISTER BIT 1 SET THEN
16677                                   ;.      ERROR WRITE BYTE ALLOCATES THE CACHE
16678                                   ;ENDIF
16679                                   ;WRITE PATTERN 1 TO TEST LOCATION
16680                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION+1
16681                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16682                                   ;.      IF B0 PARITY ERROR THEN
16683                                   ;.      .       ERROR LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16684                                   ;.      ELSE
16685                                   ;.      .       IF B1 PARITY ERROR THEN
16686                                   ;.      .       .       ERROR HIGH BYTE PARITY ERROR ON WRITE BYTE HIT
16687                                   ;.      .       ELSE
16688                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16689                                   ;.      .       ENDIF
16690                                   ;.      ENDIF
16691                                   ;ELSE
16692                                   ;.      READ TEST LOCATION
16693                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16694                                   ;.      .       IF BYTE DATA REVERSED THEN
16695                                   ;.      .       .       ERROR BYTES REVERSED ON WRITE CYCLES
16696                                   ;.      .       ELSE
16697                                   ;.      .       .       IF HIGH BYTE DATA ZERO THEN
16698                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16699                                   ;.      .       .       ELSE
16700                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16701                                   ;.      .       .       ENDIF
16702                                   ;.      .       ENDIF
16703                                   ;.      ENDIF
16704                                   ;ENDIF
16705                                   ;WRITE PATTERN 1 TO TEST LOCATION
16706                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION LOW BYTE
16707                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16708                                   ;.      IF B0 PARITY ERROR THEN
16709                                   ;.      .       ERROR LOW BYTE PARITY
16710                                   ;.      ELSE
16711                                   ;.      .       IF B1 PARITY ERROR THEN
```

```
16712                                   ;.      .       .       ERROR HIGH BYTE PRITY
16713                                   ;.      .       ELSE
16714                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16715                                   ;.      .       ENDIF
16716                                   ;.      ENDIF
16717                                   ;ELSE
16718                                   ;.      READ TEST LOCATION
16719                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16720                                   ;.      .       IF BYTE DATA REVERSED THEN
16721                                   ;.      .       .       ERROR BYTES REVERSED
16722                                   ;.      .       ELSE
16723                                   ;.      .       .       IF LOW BYTE DATA ZERO THEN
16724                                   ;.      .       .       .       ERROR IN DATA PATH
16725                                   ;.      .       .       ELSE
16726                                   ;.      .       .       .       ERROR IN DATA PATH
16727                                   ;.      .       .       ENDIF
16728                                   ;.      .       ENDIF
16729                                   ;.      ENDIF
16730                                   ;ENDIF
16731                                   ;CLEAR CACHE CONTROL REGISTER
16732                                   ;RESTORE VECTOR 114
16733                                   ;EXIT   TST
16734                                   ;
16735                                   ;BYTE PARITY ROUTINE:   SAVE MSER
16736                                   ;                       RETURN
16737                                   ;ENDTST
16738
16739                                   ;;**********************************************************************
    103172  000004                      TST7:   SCOPE
16740 103174  000240                      NOP
16741 103176  005737  003032              TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16742 103202  001002                      BNE     99$                     ; NOT YET
16743 103204  000240                      NOP                             ; DEBUG AID
16744 103206  000517                      BR      TST10           ;;GO TO NEXT TEST
16745 103210  000240                  99$:    NOP
16746 103212  013737  000114  003012      MOV     @#114,  SLOC00          ;SAVE VECTOR 114
16747 103220  012737  103434  000114      MOV     #BYPARR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
16748 103226  005003                      CLR     R3                      ;CLEAR MSER SAVE REGISTER
16749 103230  012737  000200  177746      MOV     #BIT07, CCR             ;SET PARITY ABORT BIT IN CCR
16750 103236  005737  023162              TST     TSTLOC+8192.            ;READ TEST LOCATION + 4K TO CAUSE MISS
16751 103242  042737  001000  177520      BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16752 103250  105037  003162              CLRB    TSTLOC                  ;WRITE LOW BYTE OF TEST LOCATION
16753 103254  105737  003162              TSTB    TSTLOC                  ;READ LOW BYTE OF TEST LOCATION
16754 103260  013737  177752  111464      MOV     HITMIS,RECDAT           ;STORE REGISTER
16755 103266  032737  000004  111464      BIT     #BIT02,RECDAT           ;IF BIT1 OF HIT/MISS REGISTER SET
16756 103274  001401                      BEQ     1$                      ;THEN
16757 103276  104020                      ERROR   •20                     ;WRITE BYTE ALLOCATES CACHE
16758 103300  005037  003162          1$: CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16759 103304  152737  000377  003163      BISB    #377,   @#TSTLOC+1      ;WRITE PATTERN 2 TO HIGH BYTE
16760 103312  013737  177752  111464      MOV     HITMIS,RECDAT           ;STORE REGISTER
16761 103320  032737  000004  111464      BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16762 103326  001002                      BNE     2$                      ;THEN
16763 103330  104021                  101$:   ERROR   •21                     ;WRITE BYTE HIT DOES NOT RECORD HIT
16764 103332  000405                      BR      3$                      ;ELSE
16765 103334  022737  177400  003162  2$: CMP     #177400,@#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
16766 103342  001401                      BEQ     3$                      ;PATTERN 2 THEN
16767 103344  104022                      ERROR   •22                     ;BYTES REVERSED ON WRITE CYCLES
```

TEST - BYTE ALLOCATION TEST

```
16768 103346 005037 003162          3$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16769 103352 152737 000377 003162           BISB    #377,   @#TSTLOC        ;WRITE PATTERN 2 TO LOW BYTE
16770 103360 013737 177752 111464           MOV     HITMIS,RECDAT           ;STORE REGISTER
16771 103366 032737 000004 111464           BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16772 103374 001001                         BNE     4$                      ;THEN
16773 103376 104012                         ERROR   •12                     ;LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16774 103400 022737 000377 003162   4$:     CMP     #377,   @#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
16775 103406 001401                         BEQ     5$                      ;PATTERN 2 THEN
16776 103410 104022                         ERROR   •22                     ;BYTES REVERSED ON WRITE CYCLES
16777 103412 005037 177746          5$:     CLR     CCR                     ;CLEAR CCR BEFORE EXIT
16778 103416 052737 001000 177520           BIS     #1000,@#CSR             ;ENABLE HALT ON BREAK
16779 103424 013737 003012 000114           MOV     SLOC00, @#114           ;RESTORE VECTOR 114
16780 103432 000405                         BR      TST10           ;;GO TO NEXT TEST
16781
16782
16783 103434 011637 001122          BYPARR: MOV     (SP),   @#BDADR         ;SAVE ADDRESS THAT CAUSE ABORT
16784 103440 013703 177744                  MOV     MSER,   R3              ;SAVE CONTENTS OF MSER
16785 103444 000002                         RTI                             ;RETURN
16786
```

```
                                .SBTTL  TEST  - PDR BIT15 (BYPASS) TEST
                                ;PDR BIT15 (BYPASS) TEST - THIS TEST WILL VERIFY THAT WHEN BIT<15> IS SET
                                ;IN A PDR AND AN ACCESS IS MADE TO A LOCATION MAPPED BY THE SELECTED PDR
                                ;THAT WOULD NORMALLY CAUSE A CACHE ACCESS, THE CACHE IS BYPASSED (I.E. THE
                                ;CACHE LOCATION IS INVALIDATED AND A MAIN MEMORY IS READ. THIS WILL BE DONE
                                ;BY FIRST WRITING A TEST LOCATION WITH A KNOWN PATTERN TO ALLOCATE THE ADDRESS.
                                ;THEN THE LOCATION WILL BE WRITTEN AGAIN WITH THE MMU ENABLED AND BIT <15> SET
                                ;IN THE CONTROLING PDR WITH A NEW PATTERN. THE LOCATION WILL THEN BE READ
                                ;AND A MISS SHOULD BE LOGGED IN THE HIT/MISS REGISTER. THIS READ WILL ALSO
                                ;BRING VALID DATA INTO CACHE FROM MEMORY. THE MMU WILL AGAIN BE ENABLED
                                ;WITH BIT <15> SET AND A READ CYCLE DONE. THIS SHOULD INVALIDATE THE CACHE.
                                ;NEXT THE MMU WILL BE DISABLED AND THE LOCATION READ FOR A THIRD TIME. THIS
                                ;WILL BE DONE WITH BIT<15> STILL SET. A MISS SHOULD ALSO BE LOGGED. LASTLY
                                ;BIT<15> WILL BE CLEARED AND THE MMU ENABLED AND THE LOCATION AGAIN READ.
                                ;THIS TIME A CACHE HIT SHOULD BE LOGGED.
                                ;
                                ;
                                ;BGNTST
                                ;SETUP MMU REGISTERS
                                ;WRITE TEST LOCATION WITH PATTERN 1
                                ;SET BIT<15> IN PDR FOR TEST LOCATION
                                ;ENABLE MMU
                                ;WRITE TEST LOCATION WITH PATTERN 2
                                ;DISABLE MMU AND CLEAR BIT<15> IN PDR
                                ;READ TEST LOCATION
                                ;IF HIT/MISS REGISTER BIT<1> SET THEN
                                ;.       ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
                                ;ENDIF
                                ;SET BIT<15> IN PDR FOR TEST LOCATION
                                ;ENABLE MMU
                                ;READ TEST LOCATION (SHOULD INVALIDATE CACHE)
                                ;DISABLE MMU
                                ;READ TEST LOCATION
                                ;IF HIT/MISS REGISTER BIT<1> SET THEN
                                ;.       ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
                                ;ENDIF
                                ;CLEAR BIT<15> IN PDR
                                ;TURN ON MMU
                                ;READ TEST LOCATION
                                ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
                                ;.       ERROR IN RECORDING HITS IN HIT/MISS
                                ;ENDIF
                                ;TURN OFF MMU
                                ;ENDTST
                                ;
                                ;;**********************************************************************
    103446  000004              TST10:  SCOPE
16834 103450  000240                    NOP
16835 103452  005737  003032            TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16836 103456  001002                    BNE     99$                     ; NOT YET
16837 103460  000240                    NOP                             ; DEBUG AID
16838 103462  000512                    BR      TST11           ;;GO TO NEXT TEST
16839 103464  000240              99$:  NOP
16840 103466  004737  134110            JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT REGISTERS
16841 103472  005037  003162            CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16842 103476  012737  177777  001124    MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
16843 103504  052737  100000  172300    BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
```

TEST - PDR BIT15 (BYPASS) TEST

```
16844 103512  005237  177572                 INC     SR0                     ;TURN ON MEMORY MANAGEMENT
16845 103516  012737  177777  003162         MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16846 103524  005037  177572                 CLR     SR0                     ;TURN OFF MEMORY MANAGEMENT
16847 103530  042737  100000  172300         BIC     #BIT15, KIPDR0          ;CLEAR BIT15 IN PDR FOR TEST LOCATION
16848 103536  042737  001000  177520         BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16849 103544  013701  003162                 MOV     @#TSTLOC,R1             ;READ TEST LOCATION
16850 103550  013737  177752  111464         MOV     HITMIS,RECDAT           ;SAVE HIT/MISS REGISTER
16851 103556  032737  000004  111464         BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
16852 103564  001401                         BEQ     2$                      ;THEN
16853 103566  104023                         ERROR   •23                     ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
16854 103570  052737  100000  172300  2$:    BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
16855 103576  005237  177572                 INC     SR0                     ;TURN ON MEMORY MANAGEMENT
16856 103602  005737  003162                 TST     TSTLOC                  ;READ TEST LOCATION
16857 103606  042737  100000  172300         BIC     #BIT15, KIPDR0          ;CLEAR BIT 15 IN PDR
16858 103614  005037  177572                 CLR     SR0                     ;TURN OFF MMU
16859 103620  013701  003162                 MOV     @#TSTLOC,R1             ;READ TEST LOCATION
16860 103624  013737  177752  111464         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
16861 103632  032737  000004  111464         BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
16862 103640  001401                         BEQ     3$                      ;THEN
16863 103642  104023                         ERROR   •23                     ;CONDITIONAL BYPASS DOESN'T INVALIDATE 0 CACHE
16864 103644  005237  177572          3$:    INC     SR0                     ;TURN ON MEMORY MANAGEMENT
16865 103650  005737  003162                 TST     @#TSTLOC                ;READ TEST LOCATION ONE MORE TIME
16866 103654  013737  177752  111464         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
16867 103662  032737  000004  111464         BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 NOT SET
16868 103670  001001                         BNE     4$                      ;THEN
16869 103672  104045                         ERROR   •45                     ;ERROR IN RECORDING HITS IN HIT/MISS
16870 103674  042737  000001  177572  4$:    BIC     #BIT00, SR0             ;TURN OFF MMU
16871 103702  052737  001000  177520         BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16872
```

```
16874                                   .SBTTL  TEST  - FLUSH CACHE TEST
16875                                   ;FLUSH CACHE TEST - THIS TEST WILL VERIFY THAT WHEN CCR BIT<8> IS
16876                                   ;SET, THE ENTIRE CACHE IS INVALIDATED. FIRST 8K BYTES OF ADDRESSES
16877                                   ;WILL BE READ TO ALLOCATE CACHE. THEN THE CACHE WILL BE FLUSHED.
16878                                   ;THE SAME SET OF ADDRESS WILL THEN BE READ AND THE HIT/MISS REGISTER
16879                                   ;CHECKED AFTER EACH READ FOR A MISS TO BE LOGGED.
16880                                   ;
16881                                   ;BGNTST
16882                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16883                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16884                                   ;DO UNTIL LOOP COUNTER = 0
16885                                   ;.      READ @ADDRESS+
16886                                   ;ENDDO
16887                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16888                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16889                                   ;INITIALIZE MISS COUNT TO 4KWORD COUNT
16890                                   ;FLUSH CACHE
16891                                   ;DO UNTIL LOOP COUNTER = 0
16892                                   ;.      READ @ADDRESS+
16893                                   ;.      ICREMENT ADDRESS TO READ EVERY 2WORDS
16894                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
16895                                   ;.              DECREMENT MISS COUNT
16896                                   ;.      ENDIF
16897                                   ;ENDDO
16898                                   ;GET FIRST ADDRESS OF 8KB BUFFER
16899                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16900                                   ;DO UNTIL LOOP COUNTER = 0
16901                                   ;.      READ @ADDRESS+
16902                                   ;ENDDO
16903                                   ;GET LAST ADDRESS OF 8KB BUFFER
16904                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16905                                   ;FLUSH CACHE
16906                                   ;DO UNTIL LOOP COUNTER = 0
16907                                   ;.      READ @ADDRESS-
16908                                   ;.      DECREMENT ADDRESS TO READ EVERY 2WORDS
16909                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
16910                                   ;.              DECREMENT MISS COUNT
16911                                   ;.      ENDIF
16912                                   ;ENDDO
16913                                   ;IF MISS COUNT NOT EQUAL TO 4096. THEN
16914                                   ;.      ERROR HITS RECORDED AFTER FLUSHING CACHE
16915                                   ;ENDIF
16916                                   ;ENDTST
16917                                   ;
16918                                   ;;*****************************************************************
    103710  000004                      TST11:  SCOPE
16919 103712  000240                              NOP
16920 103714  005737  003032                      TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
16921 103720  001002                              BNE     99$                     ; NOT YET
16922 103722  000240                              NOP                             ; DEBUG AID
16923 103724  000517                              BR      TST12           ;;GO TO NEXT TEST
16924 103726  000240                      99$:    NOP
16925 103730  004737  134110                      JSR     PC,     INITMM          ;INITIALIZE MMU
16926 103734  012737  001600  172354              MOV     #1600,KIPAR6            ;LET PAR6 POINT TO LOW 28K
16927 103742  012701  140000                      MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
16928 103746  005237  177572                      INC     SR0                     ;ENABLE MMU
16929 103752  012702  010000                      MOV     #4096., R2              ;INIT LOOP COUNTER TO 4K WORD COUNT
```

```
103756  005721                  1$:     TST     (R1)+                   ;READ ADDRESS TO ALLOCATE CACHE
103760  077202                          SOB     R2,     1$              ;DO UNTIL ALL LOCATIONS ALLOCATED
103762  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
103766  012702  004000                  MOV     #2048., R2              ;INIT LOOP COUNTER TO 2K WORD COUNT
103772  012703  010000                  MOV     #4096., R3              ;INITIALIZE MISS COUNT TO 4K
103776  012737  000400  177746          MOV     #400,   CCR             ;FLUSH CACHE
104004  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
104012  005721                  2$:     TST     (R1)+                   ;READ ADDRESS
104014  013737  177752  111464          MOV     @#HITMIS,RECDAT         ;STORE REGISTER
104022  032737  000004  111464          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER NOT EQUAL
104030  001401                          BEQ     3$                      ;TO ZERO THEN
104032  005303                          DEC     R3                      ;DECREMENT MISS COUNT
104034  062701  000002          3$:     ADD     #2,R1                   ;DO IN 2 WORD INCREMENTS (PMI MEMORY)
104040  077214                          SOB     R2,     2$              ;LOOP UNTIL ENTIRE CACHE CHECKED
104042  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
104050  012702  010000                  MOV     #4096.,R2               ;DO FOR 4K
104054  012701  140000                  MOV     #140000,R1              ;STARTING WITH 0
104060  005721                  4$:     TST     (R1)+                   ;ALLOCATE IN CACHE
104062  077202                          SOB     R2,4$                   ;DO FOR 4K
104064  012702  004000                  MOV     #2048.,R2               ;DO FOR THE 2ND HALF
104070  012701  160000                  MOV     #160000,R1              ;START WITH LAST LOCATION
104074  012737  000400  177746          MOV     #400,CCR                ;FLUSH CACHE
104102  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
104110  005741                  5$:     TST     -(R1)                   ;READ ADDRESS
104112  013737  177752  111464          MOV     @#HITMIS,RECDAT         ;STORE REGISTER
104120  032737  000004  111464          BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER NOT EQUAL
104126  001401                          BEQ     6$                      ;TO ZERO THEN
104130  005303                          DEC     R3                      ;DECREMENT MISS COUNT
104132  162701  000002          6$:     SUB     #2,R1                   ;DO IN 2 WORD DECREMENTS
104136  077214                          SOB     R2,5$                   ;DO FOR 2K WORDS
104140  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
104146  005037  177572                  CLR     SR0                     ;TURN OFF MMU
104152  022703  010000                  CMP     #4096., R3              ;IF MISS COUNT NOT EQUAL TO 4K WORDS
104156  001401                          BEQ     8$                      ;THEN
104160  104024                          ERROR   +24                     ;HITS RECORDED AFTER FLUSHING CACHE
104162                          8$:
104162  000400                          BR      TST12           ;;GO TO NEXT TEST
```

```
16968                                   .SBTTL  TEST  - UNCONDITIONAL BYPASS TEST
16969                                   ;UNCONDITIONAL BYPASS TEST - THIS TEST WILL VERIFY THAT WHEN CCR
16970                                   ;BIT<9> IS SET, A MEMORY REFERENCE IS FORCED TO MAIN MEMORY. THIS
16971                                   ;WILL ALSO VERIFY THAT READ AND WRITE HIT INVALIDATE THE CACHE
16972                                   ;LOCATIONS WHEN BYPASS IS SET.
16973                                   ;
16974                                   ;
16975                                   ;BGNTST
16976                                   ;
16977                                   ;READ TEST LOCATIONS TO SETUP POSSIBILITY OF HIT
16978                                   ;SET CCR BIT<9>
16979                                   ;
16980                                   ;READ FIRST TEST LOCATION
16981                                   ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
16982                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
16983                                   ;ENDIF
16984                                   ;
16985                                   ;WRITE SECOND TEST LOCATION
16986                                   ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
16987                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
16988                                   ;ENDIF
16989                                   ;CLEAR CCR BIT<9>
16990                                   ;
16991                                   ;READ FIRST LOCATION
16992                                   ;IF HIT/MISS REGISTER BIT<1> SET THEN
16993                                   ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
16994                                   ;ENDIF
16995                                   ;
16996                                   ;READ SECOND LOCATION
16997                                   ;IF HIT/MISS REGISTER BIT<1> SET THEN
16998                                   ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
16999                                   ;ENDIF
17000                                   ;
17001                                   ;ENDTST
17002                                   ;
17003                                   ;
17004
17005                                   ;;*****************************************************************
    104164  000004                      TST12:  SCOPE
17006 104166  000240                          NOP
17007 104170  005737  003032                   TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17008 104174  001002                          BNE     99$                     ; NOT YET
17009 104176  000240                          NOP                             ; DEBUG AID
17010 104200  000471                          BR      TST13           ;;GO TO NEXT TEST
17011 104202  000240                  99$:    NOP
17012
17013 104204  005737  003162                   TST     @#TSTLOC                ;ALLOCATE FIRST TEST LOCATION
17014 104210  005737  003164                   TST     @#TSTLOC+2              ;ALLOCATE SECOND TEST LOCATION
17015 104214  052737  001000  177746           BIS     #BIT09, CCR             ;SET CCR BIT 9 (CACHE BYPASS)
17016 104222  042737  001000  177520   1$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17017 104230  005737  003162                   TST     @#TSTLOC                ;READ FIRST TEST LOCATION
17018 104234  013737  177752  111464           MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17019 104242  032737  000004  111464           BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
17020 104250  001001                          BNE     2$                      ;THEN
17021 104252  104045                          ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<9>=1
17022 104254  005037  003164           2$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
17023 104260  013737  177752  111464           MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
```

TEST - UNCONDITIONAL BYPASS TEST

```
17024 104266  032737  000004  111464          BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
17025 104274  001001                          BNE     3$                      ;THEN
17026 104276  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCCR<9>=1
17027 104300  005737  003162          3$:     TST     @#TSTLOC                ;READ FIRST TEST LOCATION
17028 104304  013737  177752  111464          MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17029 104312  032737  000004  111464          BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
17030 104320  001401                          BEQ     4$                      ;THEN
17031 104322  104025                          ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
17032 104324  005037  003164          4$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
17033 104330  013737  177752  111464          MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17034 104336  032737  000004  111464          BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
17035 104344  001401                          BEQ     5$                      ;THEN
17036 104346  104025                          ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
17037 104350  042737  001000  177746  5$:     BIC     #BIT09,CCR              ;RESTORE CCR
17038 104356  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17039
```

```
17041                                   .SBTTL  TEST  - WRITE WRONG DATA PARITY TEST
17042                                   ;WRITE WRONG DATA PARITY TEST - THIS TEST WILL VERIFY THAT WHEN CCR
17043                                   ;BIT<6> = 1 AND CCR BITS<7,0> = 0,1, A READ MISS OCCURS AFTER A
17044                                   ;WRITE. THE WRITE WITH CCR BIT<6> = 1 TO A LOCATION WILL CAUSE A
17045                                   ;CACHE UPDATE AND WRONG PARITY TO BE WRITTEN SO WHEN THE LOCATION
17046                                   ;IS READ INSTEAD OF A HIT BEING RECORDED THE CACHE PARITY ERROR
17047                                   ;WILL CAUSE A CACHE MISS. THIS TEST WILL BE DONE A BYTE AT A TIME.
17048                                   ;
17049                                   ;BGNTST
17050                                   ;SAVE CONTENTS OF VECTOR 114
17051                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17052                                   ;CLEAR MSER
17053                                   ;IF MSER NOT CLEAR THEN
17054                                   ;.      ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
17055                                   ;ENDIF
17056                                   ;WRITE TEST LOCATION
17057                                   ;SET BITS<6,0> IN CCR
17058                                   ;WRITE TEST LOCATION LOW BYTE
17059                                   ;CLEAR CCR BIT<6> (WRITE WRONG DATA PARITY)
17060                                   ;INITIALIZE ERROR INDICATORS
17061                                   ;READ TEST LOCATION LOW BYTE
17062                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17063                                   ;.      PARITY ERROR DOESN'T CAUSE MISS
17064                                   ;ELSE
17065                                   ;.      IF MSER BITS <7:5> ZERO THEN
17066                                   ;.      .       PARITY ERROR DOESN'T SET MSER PROPERLY
17067                                   ;.      ELSE
17068                                   ;.      .       SET ERROR IN LOW BYTE INDICATOR
17069                                   ;.      ENDIF
17070                                   ;.      ENDIF
17071                                   ;ENDIF
17072                                   ;CLEAR MSER
17073                                   ;IF MSER NOT CLEAR THEN
17074                                   ;.      .       ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
17075                                   ;ENDIF
17076                                   ;SET CCR BIT<6>
17077                                   ;WRITE TEST LOCATION HIGH BYTE
17078                                   ;CLEAR CCR BIT<6>
17079                                   ;READ TEST LOCATION HIGH BYTE
17080                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17081                                   ;.      IF MSER BITS<7:5> NOT SET THEN
17082                                   ;.      .       PARITY ERROR DOESN'T CAUSE A MISS
17083                                   ;.      ELSE
17084                                   ;.      .       SET ERROR IN HIGH BYTE INDICATOR
17085                                   ;.      ENDIF
17086                                   ;ELSE
17087                                   ;.      IF MSER BITS <7:5> ZERO THEN
17088                                   ;.      .       PARITY ERROR DOESN'T SET MSER
17089                                   ;.      ENDIF
17090                                   ;ENDIF
17091                                   ;RESTORE CONTENTS OF VECTOR 114
17092                                   ;IF ERROR INDICATORS SET THEN
17093                                   ;.      IF ERROR IN BOTH BYTES THEN
17094                                   ;.      .       PARITY ERROR IGNORED
17095                                   ;.      ELSE
17096                                   ;.      .       IF ERROR IN LOW BYTE THEN
17097                                   ;.      .       .       LOW BYTE PARITY ERROR IGNORED
```

TEST - WRITE WRONG DATA PARITY TEST

```
17098                                   ;.      .       ELSE
17099                                   ;.      .               HIGH BYTE PARITY ERROR IGNORED
17100                                   ;.      .       ENDIF
17101                                   ;.      ENDIF
17102                                   ;ENDIF
17103                                   ;EXIT   TST
17104                                   ;
17105                                   ;DATA PARITY ABORT ROUTINE:
17106                                   ;                               ILLEGAL PARITY INTERRUPT
17107                                   ;                               RETURN
17108                                   ;
17109                                   ;ENDTST
17110                                   ;;*****************************************************************************
     104364  000004                     TST13:  SCOPE
17111 104366  000240                             NOP
17112 104370  005737  003032                     TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17113 104374  001002                             BNE     99$                     ; NOT YET
17114 104376  000240                             NOP                             ; DEBUG AID
17115 104400  000537                             BR      TST14           ;;GO TO NEXT TEST
17116 104402  000240                     99$:    NOP
17117 104404  013737  000114  003012             MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17118 104412  012737  104670  000114             MOV     #DAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
17119 104420  005037  177744                     CLR     MSER                    ;CLEAR MSER
17120 104424  005737  177744                     TST     MSER                    ;IF MSER NOT CLEAR
17121 104430  001401                             BEQ     1$                      ;THEN
17122 104432  104026                             ERROR   .26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
17123 104434  005037  003162             1$:     CLR     @#TSTLOC                ;WRITE TEST LOCATION TO ALLOCATE CACHE
17124 104440  042737  001000  177520             BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17125 104446  012737  000101  177746             MOV     #101,   CCR             ;SET BITS<6,0> IN CCR
17126 104454  112737  000377  003162             MOVB    #377,   TSTLOC          ;WRITE LOW BYTE WITH BAD PARITY
17127 104462  042737  000100  177746             BIC     #BIT06, CCR             ;CLEAR WRITE WRONG DATA PARITY BIT
17128 104470  005002                             CLR     R2                      ;CLEAR ERROR INDICATORS
17129 104472  105737  003162                     TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17130 104476  013703  177752                     MOV     HITMIS, R3              ;SAVE HIT/MISS
17131 104502  032703  000004                     BIT     #BIT02, R3              ;IF BIT 1 SET IN HIT/MISS REGISTER
17132 104506  001402                             BEQ     2$                      ;THEN
17133 104510  104027                             ERROR   .27                     ;PARITY ERROR DON'T CAUSE A MISS
17134 104512  000405                             BR      3$                      ;ELSE
17135 104514  032737  000340  177744     2$:     BIT     #340,   MSER            ;IF MSER BIT<7:5> NOT ZERO
17136 104522  001001                             BNE     3$                      ;THEN
17137 104524  104030                             ERROR   .30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
17138 104526  005037  177744             3$:     CLR     MSER                    ;CLEAR MSER
17139 104532  005737  177744                     TST     MSER                    ;IF MSER NOT CLEAR
17140 104536  001401                             BEQ     4$                      ;THEN
17141 104540  104026                             ERROR   .26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
17142 104542  052737  000100  177746     4$:     BIS     #BIT06, CCR             ;SET CCR BIT 6 (WRITE WRONG PARITY)
17143 104550  112737  000377  003163             MOVB    #377,   @#TSTLOC+1      ;WRITE HIGH BYTE OF TEST LOCATION
17144 104556  042737  000100  177746             BIC     #BIT06, CCR             ;CLEAR WRITE WRONG PARITY BIT
17145 104564  105737  003163                     TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17146 104570  013737  177752  111464             MOV     HITMIS,RECDAT           ;SAVE HIT/MISS
17147 104576  032737  000004  111464             BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
17148 104604  001402                             BEQ     5$                      ;THEN
17149 104606  104027                             ERROR   .27                     ;PARITY ERROR DON'T CAUSE A MISS
17150 104610  000405                             BR      6$                      ;ELSE
17151 104612  032737  000340  177744     5$:     BIT     #340,   MSER            ;IF BITS <7:5> ZERO
17152 104620  001001                             BNE     6$                      ;THEN
17153 104622  104030                             ERROR   .30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
```

TEST - WRITE WRONG DATA PARITY TEST

```
17154 104624  005037  177746          6$:     CLR     CCR                     ;CLEAR CCR BEFORE EXIT
17155 104630  013737  003012  000114          MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
17156 104636  032702  000003                  BIT     #3,     R2              ;IF ERROR INDICATORS SET
17157 104642  001401                          BEQ     7$                      ;THEN
17158 104644  104031                          ERROR   .+31                    ;PARITY ERROR IGNORED
17159 104646  005037  177746          7$:     CLR     CCR                     ;CLEAR CCR
17160 104652  013737  003012  000114          MOV     SLOC00,@#114            ;RESTORE PARITY TRAP
17161 104660  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17162 104666  000404                          BR      TST14           ;;GO TO NEXT TEST
17163
17164
17165 104670  011637  001122          DAPABO: MOV     (SP),   $BDADR          ;SAVE ADDRESS THAT CAUSED ABORT
17166 104674  104007                          ERROR   .+7                     ;ILLEGAL PARITY INTERRUPT
17167 104676  000002                          RTI
17168
```

```
17170                                   .SBTTL  TEST  - WRITE WRONG TAG PARITY
17171                                   ;WRITE WRONG TAG PARITY - THIS TEST WILL VERIFY THAT A READ MISS
17172                                   ;OCCURS AFTER A WRITE WILL CCR<10> = 1 AND CCR<7,0> = 0,1. THE WRITE
17173                                   ;TO THE LOCATION WILL CAUSE A CACHE UPDATE BUT THE TAG WILL BE
17174                                   ;WRITTEN WITH THE WRONG PARITY. WHEN THE LOCATION IS READ INSTEAD
17175                                   ;OF A CACHE HIT OCCURRING THE PARITY ERROR SHOULD CAUSE A CACHE
17176                                   ;MISS.
17177                                   ;
17178                                   ;BGNTST
17179                                   ;SAVE CONTENTS OF VECTOR 114
17180                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17181                                   ;CLEAR MSER
17182                                   ;SET WRITE WRONG TAG PARITY BIT<10>
17183                                   ;WRITE TEST LOCATION
17184                                   ;CLEAR CCR BIT<10> AND SET ABORT DISABLE BIT<0>
17185                                   ;READ TEST LOCATION
17186                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17187                                   ;.      IF MSER BITS <7:5> SET THEN
17188                                   ;.              PARITY ERROR DOESN'T CAUSE MISS
17189                                   ;.      ELSE
17190                                   ;.              PARITY ERROR DOESN'T SET MSER WITH CCR<7>=0
17191                                   ;.      ENDIF
17192                                   ;ENDIF
17193                                   ;SET WRITE WRONG TAG PARITY BIT<10>
17194                                   ;WRITE TO A BYTE OF A TEST LOCATION
17195                                   ;SAVE HIT/MISS REGISTER
17196                                   ;CLEAR WRITE WRONG TAG PARITY BIT
17197                                   ;IF BIT<1> NOT SET IN HIT/MISS REGISTER THEN
17198                                   ;.      ERROR
17199                                   ;ENDIF
17200                                   ;RESTORE VECTOR 114
17201                                   ;EXIT   TST
17202                                   ;
17203                                   ;TAG PARITY ABORT ROUTINE:
17204                                   ;                               ILLEGAL PARITY INTERRUPT
17205                                   ;                               RETURN
17206                                   ;ENDTST
17207                                   ;;*****************************************************************************
      104700  000004                    TST14:  SCOPE
17208 104702  000240                            NOP
17209 104704  005737  003032                    TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17210 104710  001002                            BNE     99$                     ; NOT YET
17211 104712  000240                            NOP                             ; DEBUG AID
17212 104714  000453                            BR      TST15           ;;GO TO NEXT TEST
17213 104716  000240                    99$:    NOP
17214 104720  013737  000114  003012            MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17215 104726  012737  105034  000114            MOV     #TAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
17216 104734  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17217 104742  005037  177744                    CLR     MSER                    ;CLEAR MSER
17218 104746  012737  002000  177746            MOV     #BIT10, CCR             ;SET WRITE WRONG TAG PARITY
17219 104754  005037  003162                    CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD TAG PARITY
17220 104760  012737  000001  177746            MOV     #BIT00, CCR             ;CLEAR BIT 10 AND SET BIT 0
17221 104766  005737  003162                    TST     @#TSTLOC                ;READ TEST LOCATION
17222 104772  013737  177752  111464            MOV     HITMIS,RECDAT           ;SAVE HIT/MISS REGISTER
17223 105000  032737  000004  111464            BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
17224 105006  001401                            BEQ     2$                      ;THEN
17225 105010  104027                            ERROR   +27                     ;PARITY ERROR DON'T CAUSE A MISS
```

TEST - WRITE WRONG TAG PARITY

```
17226 105012  013737  003012  000114  2$:     MOV     SLOCOO, @#114           ;RESTORE CONTENTS OF VECTOR 114
17227 105020  005037  177744                  CLR     MSER                    ;CLEAR ERRORS
17228 105024  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17229 105032  000404                          BR      TST15           ;;GO TO NEXT TEST
17230
17231
17232 105034  011637  001122          TAPABO: MOV     (SP),$BDADR             ;SAVE ADDRESS THAT CAUSE ABORT
17233 105040  104007                          ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17234 105042  000002                          RTI
17235
```

```
                                        .SBTTL  TEST  - PARITY ABORT TEST
                                        ;PARITY ABORT TEST - THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
                                        ;1,0, AN ABORT OCCURS ON THE EXECUTION OF AN INSTRUCTION THAT HAS
                                        ;BEEN WRITTEN WITH WRONG PARITY.
                                        ;
                                        ;
                                        ;BGNTST
                                        ;SAVE VECTOR 114
                                        ;SAVE VECTOR 4
                                        ;LET VECTOR 114 POINT TO ABORT ROUTINE
                                        ;LET VECTOR 4 POINT TO ERROR ROUTINE
                                        ;CLEAR EXPECTING ABORT FLAG
                                        ;SET CCR<10> WRITE WRONG TAG PARITY BIT
                                        ;WRITE TEST ADDRESS
                                        ;CLEAR CCR<10>
                                        ;SET CCR TO #200 ENABLE PARITY ABORTS
                                        ;SET EXPECTING ABORT FLAG
                                        ;READ TEST ADDRESS
                                        ;IF ABORT FLAG NE 0 THEN
                                        ;.      ERROR IN PARITY ABORT LOGIC
                                        ;ENDIF
                                        ;RESTORE VECTOR 114
                                        ;RESTORE VECTOR 4
                                        ;EXIT   TST
                                        ;
                                        ;ABORT ROUTINE: IF EXPECTING ABORT FLAG NOT SET THEN
                                        ;               .       ERROR NO ABORT SHOULD HAVE OCCURRED
                                        ;               ELSE
                                        ;               .       CLEAR (EXPECTING) ABORT FLAG
                                        ;               ENDIF
                                        ;               IF MSER NOT EQUAL TO 100040 THEN
                                        ;               .       PARITY ABORT LOGIC DOESN'T SET MSER PROPERLY
                                        ;               ENDIF
                                        ;               IF PC = UPDATED PC THEN
                                        ;               .       ILLEGAL PARITY ABORT
                                        ;               ENDIF
                                        ;               RETURN
                                        ;
                                        ;ENDTST
                                        ;;*****************************************************************
105044  000004                          TST15:  SCOPE
105046  000240                                  NOP
105050  005737  003032                          TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
105054  001002                                  BNE     99$                     ; NOT YET
105056  000240                                  NOP                             ; DEBUG AID
105060  000504                                  BR      TST16           ;;GO TO NEXT TEST
105062  000240                          99$:    NOP

105064  013737  000114  003012                  MOV     @#114,  SLOC00          ;SAVE VECTOR 114
105072  013737  000004  003014                  MOV     @#4,    SLOC01          ;SAVE VECTOR 4
105100  012737  105216  000114                  MOV     #ABORTR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
                                        ;
                                        ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
                                        ;
105106  012704  105106                          MOV     #.,R4                   ;START WITH CURRENT
105112  005724                          1$:     TST     (R4)+                   ;READ A WORD
105114  022704  105216                          CMP     #ABORTR,R4              ;GOT TO ABORT ROUTINE?
```

TEST - PARITY ABORT TEST

```
17293 105120 001374                        BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
17294 105122 005000                        CLR     R0                      ;CLEAR EXPECTING ABORT FLAG
17295 105124 042737 001000 177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17296 105132 012737 002000 177746          MOV     #BIT10, CCR             ;SET WRITE WRONG PARITY BIT
17297 105140 005037 003162                 CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH BAD PARITY
17298 105144 012737 000200 177746          MOV     #BIT07, CCR             ;ENABLE ABORTS, CLEAR WWTP BIT
17299 105152 005100                        COM     R0                      ;SET EXPECTING ABORT FLAG
17300 105154 005737 003162         ABORTI: TST     @#TSTLOC                ;READ TEST LOCATION (SHOULD CAUSE ABORT)
17301 105160 005700                        TST     R0                      ;IF ABORT FLAG NOT EQUAL ZERO
17302 105162 001401                        BEQ     1$                      ;THEN
17303 105164 104034                        ERROR   .34                     ;PARITY ABORT LOGIC DOESN'T WORK
17304 105166 052737 001000 177520  1$:     BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17305 105174 013737 003012 000114          MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17306 105202 013737 003014 000004          MOV     SLOC01, @#4             ;RESTORE VECTORE 4
17307 105210 005037 177744                 CLR     MSER
17308 105214 000426                        BR      TST16           ;;GO TO NEXT TEST
17309
17310
17311 105216 013703 177744         ABORTR: MOV     MSER,R3                 ;SAVE MSER
17312 105222 005700                        TST     R0                      ;IF EXPECTING ABORT FLAG NOT SET
17313 105224 001004                        BNE     1$                      ;THEN
17314 105226 011637 001122                 MOV     (SP),   $BDADR          ;SAVE ABORT ADDRESS
17315 105232 104007                        ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17316 105234 000401                        BR      2$                      ;ELSE
17317 105236 005000                1$:     CLR     R0                      ;CLEAR (EXPECTING) ABORT FLAG
17318 105240 022737 100040 177744  2$:     CMP     #100040,MSER            ;IF MSER NOT EQUAL TO 100040
17319 105246 001404                        BEQ     3$                      ;THEN
17320 105250 012737 100040 001124          MOV     #100040,$GDDAT          ;SAVE PROPER MSER SETTING
17321 105256 104035                        ERROR   .35                     ;PARITY ABORT DON'T SET MSER PROPERLY
17322 105260 021627 105160         3$:     CMP     (SP),   @ABORTI.4       ;IF PC EQUAL TO UPDATE PC
17323 105264 001401                        BEQ     4$                      ;THEN
17324 105266 104007                        ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17325 105270 000002                4$:     RTI                             ;RETURN
17326
```

```
                                .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
                                ;;*****************************************************************
 105272  000004                 TST16:  SCOPE
17365 105274  000240                    NOP
17366 105276  005737  003032            TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17367 105302  001002                    BNE     99$                     ; NOT YET
17368 105304  000240                    NOP                             ; DEBUG AID
17369 105306  000473                    BR      TST17           ;;GO TO NEXT TEST
17370 105310  000240            99$:    NOP
17371
17372 105312  013737  000114  003012    MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17373 105320  012737  105426  000114    MOV     #INTERR,@#114           ;LET VECTOR POINT TO INTERRUPT ROUTINE
17374                                   ;
17375                                   ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
17376                                   ;
17377 105326  012704  105326            MOV     #.,R4                   ;START WITH CURRENT
17378 105332  005724            1$:     TST     (R4)+                   ;READ A WORD
17379 105334  022704  105426            CMP     #INTERR,R4              ;GOT TO INTERRUPT ROUTINE?
17380 105340  001374                    BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
17381 105342  005001                    CLR     R1                      ;CLEAR EXPECTING INTERRUPT FLAG
17382 105344  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17383 105352  052737  000100  177746    BIS     #BIT06, CCR             ;SET WRITE WRONG DATA PARITY
```

TEST - PARITY INTERRUPT TEST

```
17384 105360  005037  003162                CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD DATA PARITY
17385 105364  005037  177746                CLR     CCR                     ;CLEAR WRITE WRONG DATA PARITY
17386 105370  005101                        COM     R1                      ;SET EXPECTING INTERRUPT FLAG
17387 105372  005737  003162        INTRPC: TST     @#TSTLOC                ;READ TEST LOCATION
17388 105376  005701                        TST     R1                      ;IF INTERRUPT FLAG NOT EQUAL ZERO
17389 105400  001401                        BEQ     1$                      ;THEN
17390 105402  104036                        ERROR   .36                     ;PARITY INTERRUPT LOGIC DOESN'T WORK
17391 105404  052737  001000  177520  1$:   BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17392 105412  013737  003012  000114        MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17393 105420  005037  177744                CLR     MSER                    ;CLEAR MSER
17394 105424  000424                        BR      TST17           ;;GO TO NEXT TEST
17395
17396
17397 105426  005701        INTERR: TST     R1                      ;IF EXPECTING INTERRUPT FLAG NOT SET
17398 105430  001004                        BNE     1$                      ;THEN
17399 105432  011637  001122                MOV     (SP),   $BDADR          ;SAVE INTERRUPT ADDRESS
17400 105436  104007                        ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17401 105440  000401                        BR      2$                      ;ELSE
17402 105442  005001                1$:     CLR     R1                      ;CLEAR (EXPECTING) INTERRUPT FLAG
17403 105444  021627  105376        2$:     CMP     (SP),   #INTRPC+4       ;IF SAVED PC NOT EQUAL TO UPDATED PC
17404 105450  001401                        BEQ     4$                      ;THEN
17405 105452  104007                        ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17406 105454  022737  000340  177744  4$:   CMP     #340,   MSER            ;IF MSER NOT EQUAL TO EXPECTED VALUE
17407 105462  001404                        BEQ     5$                      ;THEN
17408 105464  012737  000340  001124        MOV     #340,   $GDDAT          ;SAVE PROPER MSER
17409 105472  104035                        ERROR   .35                     ;PARITY INTERRUPT DON'T SET MSER PROPERLY
17410 105474  000002                5$:     RTI                             ;RETURN
17411
```

```
                                .SBTTL  TEST    - MISCELLANEOUS PARITY TEST
                                ;MISCELLANEOUS PARITY TEST - THIS TEST CHECKS THAT BYPASS CYCLES WITH
                                ;PARITY ERRORS CAUSE CACHE HIT RESPONSE AND THAT FORCE MISS CYCLES
                                ;IGNORE PARITY ERRORS.
                                ;
                                ;BGNTST
                                ;WRITE A LOCATION WITH BAD PARITY
                                ;SET BYPASS IN CCR
                                ;READ THE LOCATION BACK
                                ;IF NO HIT OR MSER NOT SET THEN
                                ;.      ERROR
                                ;ENDIF
                                ;WRITE A LOCATION WITH BAD PARITY AND SET BYPASS
                                ;IF MSER SET WHILE READING IT BACK
                                ;.      ERROR
                                ;ENDIF
                                ;ENDTST
                                ;;*****************************************************************
    105476  000004              TST17:  SCOPE
17431 105500  000240                    NOP
17432 105502  005737  003032            TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17433 105506  001002                    BNE     99$                     ; NOT YET
17434 105510  000240                    NOP                             ; DEBUG AID
17435 105512  000503                    BR      TST20           ;;GO TO NEXT TEST
17436 105514  000240              99$:  NOP
17437
17438 105516  012703  105516            MOV     #.,R3                           ;START WITH CURRENT INSTRUCTION
17439 105522  005723              10$:  TST     (R3)+                           ;READ A WORD
17440 105524  022703  105674            CMP     #4$,R3                          ;LAST WORD?
17441 105530  001374                    BNE     10$                             ;IF NOT, CONTINUE
17442                           ;
17443                           ; CHECK BYPASS AND BAD PARITY
17444                           ;
17445 105532  013737  000114  003012      MOV     @#114,  SLOC00                  ;SAVE PARITY VECTOR
17446 105540  012737  105604  000114      MOV     #1$,@#114                       ;POINT NEW VECTOR
17447 105546  042737  001000  177520      BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
17448 105554  052737  002101  177746      BIS     #BIT10!BIT06!BIT00,CCR          ;DATA AND TAG PAR., NO INT.
17449 105562  005037  003162            CLR     @#TSTLOC                        ;WRITE CYCLE
17450 105566  012737  001000  177746      MOV     #BIT09,CCR                      ;SET BYPASS
17451 105574  005737  003162            TST     @#TSTLOC                        ;BYPASS WITH WRONG PARITY
17452 105600  104045                    ERROR   .45                             ;ERROR
17453 105602  000407                    BR      2$                              ;
17454 105604  062706  000004      1$:   ADD     #4,SP                           ;ADJUST STACK
17455 105610  032737  000340  177744      BIT     #340,MSER                       ;MSER OK?
17456 105616  001001                    BNE     2$                              ;IF YES, BRANCH
17457 105620  104042                    ERROR   .42                             ;BYPASS WRONG
17458                           ;
17459                           ; CHECK FORCE MISS AND BAD PARITY
17460                           ;
17461 105622  005037  177744      2$:   CLR     MSER                            ;CLEAR MSER
17462 105626  005037  177746            CLR     CCR                             ;CLEAR CCR
17463 105632  012737  105666  000114      MOV     #3$,@#114                       ;POINT NEW VECTOR
17464 105640  052737  002101  177746      BIS     #BIT10!BIT06!BIT00,CCR          ;DATA AND TAG PAR., NO INTER
17465 105646  005037  003162            CLR     @#TSTLOC                        ;FORCE MISS WITH PARITY
17466 105652  012737  000014  177746      MOV     #14,CCR                         ;FORCE MISS
17467 105660  005737  003162            TST     @#TSTLOC                        ;ALLOCATE CACHE
17468 105664  000403                    BR      4$
```

```
105666  104042                  3$:     ERROR   +42
105670  062706  000004                  ADD     #4,SP                   ;ADJUST STACK
105674  005037  177746          4$:     CLR     CCR                     ;CLEAR CCR
105700  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
105706  013737  003012  000114          MOV     SLOC00,@#114            ;RESTORE PARITY VECTOR
105714  012737  000400  177746          MOV     #BIT08,CCR              ;FLUSH CACHE
```

```
                        .SBTTL  TEST  - MEMORY SYSTEM ERROR REGISTER TEST
                        ;MEMORY SYSTEM ERROR REGISTER TEST - THIS TEST WILL VERIFY THE
                        ;FUNCTIONALITY OF BITS <15> AND <7:5> OF THE MEMORY SYSTEM ERROR
                        ;REGISTER. THIS TEST WILL USE THE WRITE WRONG PARITY BITS (BITS<10>
                        ;AND <6> OF THE CCR) TO WRITE BAD PARITY INTO A LOCATION. THE
                        ;LOCATION WILL THEN BE READ WITH CACHE TRAPS ENABLED AND THE MSER
                        ;WILL BE CHECKED AFTER THE ABORT FOR THE CORRECT BIT(S) BEING SET.
                        ;THIS WILL BE DONE FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR WORD
                        ;ACCESSES THEN REPEATED FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR
                        ;BYTE ACCESSES. THE TEST WILL THEN BE REPEATED A THIRD TIME FOR BYTE
                        ;ACCESSES AND ABORTS DISABLED. THE MSER SHOULD CONTAIN BITS <7:5>
                        ;SET TO 1'S AND BIT <15> A ZERO FOR ALL COMBINATIONS.
                        ;
                        ;BGNTST
                        ;SAVE CONTENTS OF VECTOR 114
                        ;SETUP VECTOR TO POINT TO ABORT ROUTINE
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL WORD COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,6> FROM CCR
                        ;.      SETUP CCR FOR ABORT
                        ;.      READ TEST LOCATION                      ;THIS COULD CAUSE TRAP
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.      .       ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      UPDATE LOOP COUNTER
                        ;ENDDO
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,6> FROM CCR
                        ;.      SETUP CCR FOR ABORT
                        ;.      READ LOW BYTE TEST LOCATION
                        ;.      GET EXPECTED BYTE DATA FROM TABLE
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.      .       ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      INITIALIZE MSER
                        ;.      READ HIGH BYTE TEST LOCATION
                        ;.      GET EXPECTED BYTE DATA FROM TABLE
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.      .       ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      INCREMENT LOOP COUNTER
                        ;ENDDO
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,ö> FROM CCR
                        ;.      SETUP CCR FOR NO ABORTS
                        ;.      READ LOW BYTE TEST LOCATION
```

TEST - MEMORY SYSTEM ERROR REGISTER TEST

```
17534                                   ;.      IF RECEIVED DATA NE TO #340 THEN
17535                                   ;.      .       ERROR IN SETTING MSER
17536                                   ;.      ENDIF
17537                                   ;.      INITIALIZE MSER
17538                                   ;.      READ HIGH BYTE TEST LOCATION
17539                                   ;.      IF RECEIVED DATA NE #340 THEN
17540                                   ;.      .       ERROR IN SETTING MSER
17541                                   ;.      ENDIF
17542                                   ;.      INCREMENT LOOP COUNTER
17543                                   ;ENDDO
17544                                   ;EXIT   TST
17545                                   ;
17546                                   ;CCR TABLE:     0
17547                                   ;               100
17548                                   ;               2000
17549                                   ;               2100
17550                                   ;EXPECTED WORD DATA:    0
17551                                   ;                       100300
17552                                   ;                       100040
17553                                   ;                       100340
17554                                   ;EXPECTED BYTE DATA:    0
17555                                   ;                       0
17556                                   ;                       100100
17557                                   ;                       100200
17558                                   ;                       100040
17559                                   ;                       100040
17560                                   ;                       100140
17561                                   ;                       100240
17562                                   ;ABORT ROUTINE: RTI
17563                                   ;
17564                                   ;ENDTST
17565                                   ;;*****************************************************************************
    105722  000004                      TST20:  SCOPE
17566 105724  000240                              NOP
17567 105726  005737  003032                      TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17568 105732  001003                      BNE     99$                             ; NOT YET
17569 105734  000240                      NOP                                     ; DEBUG AID
17570 105736  000137  106424              JMP     10$                             ; YES SKIP THIS
17571 105742  000240                      99$:    NOP
17572
17573 105744  013737  000114  003012      MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17574 105752  012737  106466  000114      MOV     #ABROUT,@#114           ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17575 105760  012704  000004              MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17576 105764  012700  106426              MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17577 105770  012701  106436              MOV     #EXPWDT,R1              ;GET ADDRESS OF EXPECTED DATA TABLE
17578 105774  042737  001000  177520      BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17579 106002  005037  177744              1$:     CLR     MSER            ;INITIALIZE MSER DATA
17580 106006  012037  177746              MOV     (R0)+,  CCR             ;SETUP CCR FROM CCR TABLE
17581 106012  005037  003162              CLR     @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17582 106016  012737  000200  177746      MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17583 106024  005737  003162              TST     @#TSTLOC                ;READ TEST LOCATION
17584 106030  021137  177744              CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17585 106034  001403                      BEQ     2$                      ;DATA THEN
17586 106036  011137  001124              MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17587 106042  104035                      ERROR   +35                     ;MSER NOT SET PROPERLY
17588 106044  005721                      2$:     TST     (R1)+           ;INCREMENT POINTER THRU MSER TABLE
17589 106046  005737  023162              TST     @#TSTLOC+8192.          ;TO INSURE MISS ON THE NEXT LOOP
```

TEST - MEMORY SYSTEM ERROR REGISTER TEST

```
17590 106052  077425                        SOB     R4,     1$              ;LOOP UNTIL ALL COMBINATIONS CHECKED
17591 106054  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17592
17593                               ;CHECK BYTE OPERATIONS NOW
17594 106062  012704  000004                MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17595 106066  012700  106426                MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17596 106072  012701  106446                MOV     #EXPBDT,R1              ;GET ADDRESS OF EXPECTED BYTE DATA TABLE
17597 106076  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17598 106104  005037  177744          3$:   CLR     MSER                    ;INITIALIZE MSER
17599 106110  005737  003162                TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17600 106114  011037  177746                MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17601 106120  105037  003162                CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17602 106124  012737  000200  177746        MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17603 106132  105737  003162                TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17604 106136  021137  177744                CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17605 106142  001403                        BEQ     4$                      ;THEN
17606 106144  011137  001124                MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17607 106150  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17608 106152  005721          4$:   TST     (R1)+                           ;INCREMENT POINTER THRU MSER TABLE
17609 106154  005037  177744                CLR     MSER                    ;INITIALIZE MSER
17610 106160  005737  003162                TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17611 106164  012037  177746                MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17612 106170  105037  003163                CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17613 106174  012737  000200  177746        MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17614 106202  105737  003163                TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17615 106206  021137  177744                CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17616 106212  001403                        BEQ     5$                      ;THEN
17617 106214  011137  001124                MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17618 106220  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17619 106222  005721          5$:   TST     (R1)+                           ;INCREMENT POINTER THRU MSER TABLE
17620 106224  077451                        SOB     R4,     3$              ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17621 106226  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17622                               ;REPEAT WITHOUT ABORT
17623 106234  012704  000003                MOV     #3,     R4              ;INITIALIZE LOOP COUNTER
17624 106240  012700  106430                MOV     #CCRTBL+2,R0            ;GET ADDRESS OF CCR TABLE
17625 106244  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17626 106252  005037  177744          6$:   CLR     MSER                    ;INITIALIZE MSER
17627 106256  005737  003162                TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17628 106262  011037  177746                MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17629 106266  105037  003162                CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17630 106272  012737  000001  177746        MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17631 106300  105737  003162                TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17632 106304  022737  000340  177744        CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17633 106312  001404                        BEQ     7$                      ;THEN
17634 106314  012737  000340  001124        MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17635 106322  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17636 106324  005037  177744          7$:   CLR     MSER                    ;INITIALIZE MSER
17637 106330  005737  003162                TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17638 106334  012037  177746                MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17639 106340  105037  003163                CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17640 106344  012737  000001  177746        MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17641 106352  105737  003163                TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17642 106356  022737  000340  177744        CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17643 106364  001404                        BEQ     8$                      ;THEN
17644 106366  012737  000340  001124        MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17645 106374  104035                        ERROR   +35                     ;MSER NOT SET PROPERLY
17646 106376  077453          8$:   SOB     R4,     6$                      ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
```

TEST - MEMORY SYSTEM ERROR REGISTER TEST

```
17647 106400  005037  177744                CLR     MSER                    ;CLEAR ERROR REGISTER
17648 106404  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17649 106412  013737  003012  000114        MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17650 106420  005037  177746                CLR     CCR                     ;CLEAR ALL BIT IN CCR
17651 106424                        10$:
      106424  000421                        BR      TST21           ;;GO TO NEXT TEST
17652
17653
17654 106426  000000                CCRTBL: .WORD   0
17655 106430  000100                        .WORD   100
17656 106432  002000                        .WORD   2000
17657 106434  002100                        .WORD   2100
17658
17659 106436  000000                EXPWDT: .WORD   0
17660 106440  100300                        .WORD   100300
17661 106442  100040                        .WORD   100040
17662 106444  100340                        .WORD   100340
17663
17664 106446  000000                EXPBDT: .WORD   0
17665 106450  000000                        .WORD   0
17666 106452  100100                        .WORD   100100
17667 106454  100200                        .WORD   100200
17668 106456  100040                        .WORD   100040
17669 106460  100040                        .WORD   100040
17670 106462  100140                        .WORD   100140
17671 106464  100240                        .WORD   100240
17672
17673 106466  000002                ABROUT: RTI
17674
17675
```

```
17677                                   .SBTTL  TEST  - CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT
17678                                   ;CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT - THIS TEST WILL
17679                                   ;VERIFY THAT IF A PARITY ERROR OCCURS ON THE SAME ADDRESS REFERENCE AS A
17680                                   ;NON-EXISTENT MEMORY ERROR THAT THE CACHE DATA PATH GATE ARRAY BLOCKS THE
17681                                   ;PARITY ERROR TO THE J-11 CHIP SET. THIS WILL BE DONE BY USING THE DIAGNOSTIC
17682                                   ;BIT TO CAUSE A PARITY ERROR IN A CACHE REFERENCE THAT DOES NOT HAVE A
17683                                   ;CORRESPONDING ADDRESS IN MAIN MEMORY. THE ADDRESS WILL THEN BE READ CAUSING
17684                                   ;BOTH A CACHE PARITY ERROR AND A NON-EXISTENT MEMORY ERROR. TO AVOID HAVING
17685                                   ;TO SIZE THE ENTIRE MEMORY TO FIND A NON-EXISTENT MEMORY ADDRESS THIS TEST
17686                                   ;WILL USE THE LARGEST NON-I/O ADDRESS (17757776). THE TEST WILL FIRST READ
17687                                   ;THIS ADDRESS AND IF A NXM TRAP OCCURS THE TEST WILL BE DONE. IF THE ACCESS
17688                                   ;TO THIS ADDRESS DOES NOT TRAP THEN THE TEST WILL BE SKIPPED.
17689                                   ;
17690                                   ;BGNTST
17691                                   ;SAVE CONTENTS OF VECTOR 4
17692                                   ;SAVE CONTENTS OF VECTOR 114
17693                                   ;LET VECTOR 4 POINT TO CONTINUE TESTING (A:)
17694                                   ;LET PAR6 = #177400
17695                                   ;ACCESS ADDRESS 157776 (PHYSICAL 17757776)
17696                                   ;IF NO TRAP THEN
17697                                   ;.      GOTO ENDTST
17698                                   ;ENDIF
17699                                   ;A:
17700                                   ;SET DIAGNOSTIC AND WRITE WRONG PARITY BITS IN CCR
17701                                   ;WRITE ADDRESS 157776
17702                                   ;CLEAR CCR
17703                                   ;SET PARITY ERROR ABORT BIT IN CCR
17704                                   ;LET VECTOR 4 POINT TO CONTINUE TESTING (B:)
17705                                   ;LET VECTOR 114 POINT TO NXM-PARITY ERROR ROUTINE
17706                                   ;READ ADDRESS 157776
17707                                   ;IF NO TRAP THEN
17708                                   ;.      ERROR IN ABORT LOGIC
17709                                   ;ENDIF
17710                                   ;B:
17711                                   ;CLEAR CCR
17712                                   ;RESTORE CONTENTS OF VECTOR 114
17713                                   ;RESTORE CONTENTS OF VECTOR 4
17714                                   ;EXIT   TST
17715                                   ;
17716                                   ;
17717                                   ;NXM-PARITY ERROR ROUTINE:      RESET STACK AFTER TRAP
17718                                   ;                                       PARITY ABORT NOT BLOCKED BY NXM
17719                                   ;                               GOTO B:
17720                                   ;ENDTST
17721                                   ;;**********************************************************************
      106470  000004                    TST21:  SCOPE
17722 106472  000240                            NOP
17723 106474  005737  003032                    TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17724 106500  001002                            BNE     99$                     ; NOT YET
17725 106502  000240                            NOP                             ; DEBUG AID
17726 106504  000475                            BR      TST22           ;;GO TO NEXT TEST
17727 106506  000240                    99$:    NOP
17728
17729 106510  013737  000004  003012            MOV     @#4,    SLOC00          ;SAVE CONTENTS OF VECTOR 4
17730 106516  013737  000114  003014            MOV     @#114,  SLOC01          ;SAVE CONTENTS OF VECTOR 114
17731 106524  012737  106552  000004            MOV     #1$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
17732 106532  012737  177400  172354            MOV     #177400,KIPAR6          ;LET PAR6 = OFFSET TO HIGHEST MEMORY
```

TEST - CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABO

```
17733 106540  005237  177572                INC     SR0                     ;TURN ON MMU
17734 106544  005737  157776                TST     @#157776                ;ACCESS ADDRESS 17757776
17735 106550  000431                        BR      ABOEXT                  ;IF NO TRAP SKIP TEST
17736 106552  062706  000004          1$:   ADD     #4,     SP              ;RESET STACK AFTER TRAP
17737 106556  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17738 106564  012737  000102  177746        MOV     #102,   CCR             ;SET DIAG. AND WRITE WRONG PARITY BITS
17739 106572  005037  157776                CLR     @#157776                ;WRITE TO ADDRESS 17757776
17740 106576  012737  000200  177746        MOV     #200,   CCR             ;CLEAR CCR AND SET PARITY ABORT BIT
17741 106604  012737  106630  000004        MOV     #2$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
17742 106612  012737  106670  000114        MOV     #NXMPAR,@#114           ;LET VECTOR 114 POINT TO ERROR ROUTINE
17743 106620  005737  157776                TST     @#157776                ;READ ADDRESS 17757776 (SHOULD TRAP)
17744 106624  104037                        ERROR   +37                     ;NXM AND PARITY ABORT DIN'T HAPPEN
17745 106626  000402                        BR      ABOEXT                  ;GO EXIT TEST
17746 106630  062706  000004          2$:   ADD     #4,     SP              ;RESET STACK AFTER TRAP
17747 106634  005037  177746        ABOEXT: CLR     CCR                     ;CLEAR CCR FOR EXIT
17748 106640  005037  177572                CLR     SR0                     ;DISABLE MMU
17749 106644  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17750 106652  013737  003012  000004        MOV     SLOC00, @#4             ;RESTORE VECTOR 4
17751 106660  013737  003014  000114        MOV     SLOC01, @#114           ;RESTORE VECTOR 114
17752 106666  000404                        BR      TST22           ;;GO TO NEXT TEST
17753
17754
17755
17756 106670  062706  000004        NXMPAR: ADD     #4,     SP              ;RESET STACK AFTER TRAP
17757 106674  104040                        ERROR   +40                     ;PARITY ABORT NOT BLOCKED BY NXM TRAP
17758 106676  000002                        RTI
17759
```

```
                                        .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
                                        ;.      ERRGR 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

                                        ;;**********************************************************************
 106700  000004                 TST22:  SCOPE
 106702  000240                         NOP
 106704  005737  003032                 TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
 106710  001002                         BNE     99$                     ; NOT YET
 106712  000240                         NOP                             ; DEBUG AID
 106714  000477                         BR      TST23           ;;GO TO NEXT TEST
 106716  000240                 99$:    NOP

 106720  042737  001000  177520       BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 106726  005737  003162                 TST     TSTLOC                  ;READ TEST LOCATION TO ALLOCATE CACHE
                                        ;       TSTSET  TSTLOC                  ;DO TSTSET INSTRUCTION
 106732  007237                 7$:     .WORD   7237                    ;THESE NEXT TWO LOCATIONS ARE THE TSTSET
 106734  003162                         .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
 106736  013737  177752  111464       MOV     HITMIS,RECDAT           ;STORE REGISTER
 106744  032737  000010  111464       BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
 106752  001001                         BNE     1$                      ;THEN
 106754  104045                         ERROR   ^45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 106756  032737  000004  111464 1$:     BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
 106764  001404                         BEQ     2$                      ;THEN
 106766  013737  106732  001126       MOV     @#7$,   $BDDAT          ;SAVE OPCODE
 106774  104041                         ERROR   ^41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MIS
S
                                        ;2$:    WRTLCK  TSTLOC                  ;DO WRTLCK INSTRUCTION
 106776  007337                 2$:     .WORD   7337                    ;THESE NEXT TWO WORDS ARE THE WRTLCK
 107000  003162                         .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
 107002  013737  177752  111464       MOV     HITMIS,RECDAT           ;STORE REGISTER
 107010  032737  000010  111464       BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
```

TEST - MULTIPROCCESSING INSTRUCTION TESTS

```
  17817 107016  001001                              BNE     3$                      ;THEN
  17818 107020  104045                              ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
  17819 107022  032737  000004  111464  3$:         BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
  17820 107030  001404                              BEQ     4$                      ;THEN
  17821 107032  013737  106776  001126              MOV     #02$,   $BDDAT          ;SAVE OPCODE
  17822 107040  104041                              ERROR   .41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'TT CAUSE MIS
S
  17823 107042  106237  003162          4$:         ASRB    TSTLOC                  ;THE THIRD BYPASS INSTRUCTION
  17824 107046  013737  177752  111464              MOV     HITMIS,RECDAT           ;STORE REGISTER
  17825 107054  032737  000010  111464              BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
  17826 107062  001001                              BNE     5$                      ;THEN
  17827 107064  104045                              ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
  17828 107066  052737  001000  177520  5$:         BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
  17829 107074  032737  000004  111464              BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
  17830 107102  001404                              BEQ     TST23           ;;EXIT TEST
  17831 107104  013737  107042  001126              MOV     #04$,   $BDDAT          ;SAVE OPCODE
  17832 107112  104041                              ERROR   .41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'TT CAUSE MIS
S
  17833
  17834
  17835
```

```
                                .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
                                ;
                                ;;*****************************************************************
107114  000004                  TST23:  SCOPE
17890 107116  000240                    NOP
17891 107120  005737  003032            TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
17892 107124  001002                    BNE     99$                     ; NOT YET
```

```
107126  000240                          NOP                                 ; DEBUG AID
107130  000550                          BR      TST24                ;;GO TO NEXT TEST
107132  000240                  99$:    NOP

107134  004737  134110                  JSR     PC,     INITMM              ;SETUP MMU REGISTERS
107140  012737  002000  172354          MOV     #2000,  KIPAR6              ;LET PAR6 MAP TO ADDRESS 200000
107146  005237  177572                  INC     SR0                         ;TURN ON MEMORY MANAGEMENT
107152  012702  140000                  MOV     #140000,R2                  ;GET FIRST ADDRESS OF 4K WORD BUFFER
107156  005001                          CLR     R1                          ;INIT DATA TO BE WRITTEN
107160  052737  000002  177746          BIS     #BIT01, CCR                 ;SET DIAG BIT IN CASE NO MEMORY PRESENT
107166  042737  001000  177520          BIC     #1000,BCSR                  ;DISABLE HALT ON BREAK
107174  020127  020000          1$:     CMP     R1,     #20000              ;DO UNTIL ALL ADDRESSES WRITTEN
107200  001404                          BEQ     2$                          ;IF DONE GO CHECK DATA
107202  010122                          MOV     R1,     (R2)+               ;WRITE DATA TO ADDRESS
107204  062701  000002                  ADD     #2,     R1                  ;UPDATE DATA BY 2 (WORD BOUNDARY)
107210  000771                          BR      1$                          ;ENDDO
107212  042737  000002  177746  2$:     BIC     #BIT01, CCR                 ;CLEAR DIAGNOSTIC BIT
107220  012702  140000                  MOV     #140000,R2                  ;GET FIRST ADDRESS OF 4K WORD BUFFER
107224  005001                          CLR     R1                          ;INIT DATA TO BE CHECKED
107226  020127  020000          3$:     CMP     R1,     #20000              ;DO UNTIL ALL ADDRESSES CHECKED
107232  001426                          BEQ     5$                          ;IF DONE GO DO BYTES
107234  010137  001122                  MOV     R1,     $BDADR              ;STORE FOR ERRORS
107240  052737  100000  001122          BIS     #BIT15, $BDADR              ;
107246  012237  111464                  MOV     (R2)+,  RECDAT              ;READ TEST LOCATION
107252  013737  177752  001160          MOV     HITMIS, $TMP0               ;STORE REGISTER
107260  032737  000020  001160          BIT     #BIT04,$TMP0                ;HIT?
107266  001001                          BNE     100$                        ;IF YES, BRANCH
107270  104045                          ERROR   +45                         ;DATA RAM ERROR
107272  020137  111464          100$:   CMP     R1,     RECDAT              ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
107276  001401                          BEQ     4$                          ;THEN
107300  104043                          ERROR   +43                         ;DATA RAM ERROR
107302  062701  000002          4$:     ADD     #2,     R1                  ;UPDATE EXPECTED DATA BY 2
107306  000747                          BR      3$                          ;ENDDO
107310  012702  140000          5$:     MOV     #140000,R2                  ;GET FIRST ADDRESS OF 4K WORD BUFFER
107314  005001                          CLR     R1                          ;INIT DATA TO BE WRITTEN
107316  052737  000002  177746          BIS     #BIT01, CCR                 ;SET DIAG BIT IN CASE NO MEMORY PRESENT
107324  020127  020000          6$:     CMP     R1,     #20000              ;DO UNTIL ALL ADDRESSES WRITTEN
107330  001404                          BEQ     7$                          ;IF DONE GO CHECK DATA
107332  110122                          MOVB    R1,     (R2)+               ;WRITE DATA TO ADDRESS
107334  062701  000001                  ADD     #1,     R1                  ;UPDATE DATA BY 1 (BYTE BOUNDARY)
107340  000771                          BR      6$                          ;ENDDO
107342  042737  000002  177746  7$:     BIC     #BIT01, CCR                 ;CLEAR DIAGNOSTIC BIT
107350  012702  140000                  MOV     #140000,R2                  ;GET FIRST ADDRESS OF 4K WORD BUFFER
107354  005001                          CLR     R1                          ;INIT DATA TO BE CHECKED
107356  020127  020000          8$:     CMP     R1,     #20000              ;DO UNTIL ALL ADDRESSES CHECKED
107362  001426                          BEQ     10$                         ;IF DONE EXIT TEST
107364  010137  001122                  MOV     R1,     $BDADR              ;STORE FOR ERRORS
107370  052737  100000  001122          BIS     #BIT15, $BDADR              ;
107376  112237  111464                  MOVB    (R2)+,  RECDAT              ;READ TEST LOCATION
107402  013737  177752  001160          MOV     HITMIS, $TMP0               ;STORE REGISTER
107410  032737  000020  001160          BIT     #BIT04,$TMP0                ;HIT?
107416  001001                          BNE     101$                        ;IF YES, BRANCH
107420  104045                          ERROR   +45                         ;DATA RAM ERROR
107422  120137  111464          101$:   CMPB    R1,     RECDAT              ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
107426  001401                          BEQ     9$                          ;THEN
107430  104043                          ERROR   +43                         ;DATA RAM ERROR
107432  062701  000001          9$:     ADD     #1,     R1                  ;UPDATE EXPECTED DATA BY 1
```

TEST - DATA STORE RAM TESTS

```
17950 107436  000747                          BR      8$                      ;ENDDO
17951 107440  052737  001000  177520  10$:    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17952 107446  005037  177572                  CLR     SR0                     ;TURN OFF MMU
17953
```

```
17955                           .SBTTL  TEST  - TAG STORE RAM TESTS
17956                           ;TAG STORE RAM TESTS - THERE ARE TWO TESTS FOR THE TAG STORE RAMS. THE FIRST
17957                           ;TEST IS AN ADDRESSING TEST TO ENSURE NO DUAL ADDRESSING OF THE TAG STORE.
17958                           ;THE SECOND TEST IS A "MOVING INVERSIONS" TEST THAT CHECKS THE DATA RELIABILITY
17959                           ;OF THE RAM STORE.
17960                           ;
17961                           ;DUAL ADDRESSING TEST - THIS WILL BE DONE BY FIRST FLUSHING THE CACHE, THEN
17962                           ;THE FIRST 512(10) LOCATIONS (BLOCK 0) WILL BE WRITTEN WITH ADDRESSES THAT
17963                           ;WILL CAUSE A INCREMENTING PATTERN TO BE STORED IN THOSE LOCATIONS OF THE TAG
17964                           ;STORE RAM. NEXT THE ENTIRE 4K ADDRESS RANGE WILL BE READ. THE HIT/MISS
17965                           ;REGISTER SHOULD ONLY REPORT HITS ON THE ADDRESSES THAT WERE ALLOCATED. THIS
17966                           ;PROCESS WILL BE REPEATED FOR EACH BLOCK.
17967                           ;
17968                           ;INITIALIZE LOW ADDRESS
17969                           ;SETUP AND ENABLE MMU
17970                           ;INITIALIZE BLOCK COUNTER
17971                           ;DO UNTIL ALL BLOCKS TESTED (8 TIMES)
17972                           ;.      FLUSH CACHE AND SET DIAGNOSTIC BIT
17973                           ;.      LET CURRENT ADDRESS = LOW ADDRESS
17974                           ;.      INITIALIZE ALLOCATION COUNTER
17975                           ;.      DO UNTIL ENTIRE BLOCK ALLOCATED (1000 TIMES)
17976                           ;.      .       .       READ CURRENT ADDRESS
17977                           ;.      .       ELSE
17978                           ;.      .       .       WRITE CURRENT ADDRESS
17979                           ;.      .       ENDIF
17980                           ;.      .       UPDATE CURRENT ADDRESS (ALSO ADD 200 TO PAR)
17981                           ;.      .       FOR I/O PAGE WRITE THE SAME ADDRESS AS BEFORE
17982                           ;.      ENDDO
17983                           ;.      INITIALIZE GOOD ADDRESS
17984                           ;.      INITIALIZE CURRENT ADDRESS
17985                           ;.      INITIALIZE LOCATION COUNTER
17986                           ;.      SAVE CONTENTS OF VECTOR 114
17987                           ;.      LET VECTOR 114 POINT TO TAG STORE PARITY ABORT ROUTINE
17988                           ;.      DO UNTIL ALL LOCATIONS CHECKED (1000 TIMES)
17989                           ;.      .       INITIALIZE CHECK COUNTER
17990                           ;.      .       DO UNTIL ADDRESS IN EACH BLOCK CHECKED
17991                           ;.      .       .       READ CURRENT ADDRESS
17992                           ;.      .       .       IF HIT THEN
17993                           ;.      .       .       .       IF CURRENT ADDRESS NE GOOD ADDRESS THEN
17994                           ;.      .       .       .       .       ERROR
17995                           ;.      .       .       .       ENDIF
17996                           ;.      .       .       ELSE
17997                           ;.      .       .       .       IF CURRENT ADDRESS EQUAL GOOD ADDRESS THEN
17998                           ;.      .       .       .       .       ERROR
17999                           ;.      .       .       .       ENDIF
18000                           ;.      .       .       ENDIF
18001                           ;.      .       .       UPDATE GOOD ADDRESS (ADD #2)
18002                           ;.      .       .       UPDATE CURRENT ADDRESS
18003                           ;.      .       ENDDO
18004                           ;.      .       UPDATE LOW ADDRESS (ADD #2000)
18005                           ;.      ENDDO
18006                           ;ENDDO
18007                           ;DISABLE MMU
18008                           ;RESTORE VECTOR 114
18009                           ;ENDTST
18010
18011                           ;;*****************************************************************
```

TEST - TAG STORE RAM TESTS

```
        107452  000004                  TST24:  SCOPE
 18012  107454  000240                          NOP
 18013  107456  005737  003032                  TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
 18014  107462  001003                          BNE     99$                     ; NOT YET
 18015  107464  000240                          NOP                             ; DEBUG AID
 18016  107466  000137  110224                  JMP     18$                     ; YES SKIP THIS
 18017  107472  000240                  99$:    NOP
 18018
 18019  107474  013737  000004  001160          MOV     @#4,$TMPO               ;STORE TIMEOUT VECTOR
 18020  107502  012737  110226  000004          MOV     #20$,@#4                ;POINT NEW
 18021  107510  012737  140000  003016          MOV     #140000,LOWADD          ;INITIALIZE LOW ADDRESS (USE PAR6)
 18022  107516  004737  134110                  JSR     PC,     INITMM          ;INITIALIZE MMU
 18023  107522  005237  177572                  INC     SR0                     ;ENABLE MMU
 18024  107526  012737  000020  172516          MOV     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
 18025  107534  012737  177770  003154          MOV     #-10.,  LOOPIN          ;DO UNTIL ALL BLOCKS TESTED
 18026  107542  012701  000000          1$:     MOV     #0,R1                   ;DO IN 2 WORDS TO AVOID PMI MEMORY
 18027  107546  042737  001000  177520  9$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 18028  107554  005037  172354                  CLR     KIPAR6                  ;SET UP PAR6 FOR THIS TEST
 18029  107560  012737  000402  177746          MOV     #402,   CCR             ;FLUSH CACHE AND SET DIAG BIT
 18030  107566  013737  003016  111504          MOV     LOWADD, CURADD          ;GET FIRST ADDRESS IN CURRENT BLOCK
 18031  107574  012737  177400  003152          MOV     #-400,  ALLCTR          ;DO UNTIL ALL ADDRESSES ALLOCATED
 18032  107602  022737  002000  172354  2$:     CMP     #2000,  KIPAR6          ;IF ADDRESS LESS THAN 32K
 18033  107610  002413                          BLT     3$                      ;THEN
 18034  107612  052737  100000  172314          BIS     #BIT15,KIPDR6           ;SET BYPASS
 18035  107620  017702  001660                  MOV     @CURADD,R2              ;STORE CURRENT DATA
 18036  107624  042737  100000  172314          BIC     #BIT15,KIPDR6           ;ALLOCATE NEXT ACCESS
 18037  107632  010277  001646                  MOV     R2,@CURADD              ;WRITE ALLOCATE
 18038  107636  000402                          BR      4$                      ;ELSE
 18039  107640  005077  001640          3$:     CLR     @CURADD                 ;WRITE CURRENT ADDRESS
 18040  107644  062737  000002  111504  4$:     ADD     #2,     CURADD          ;UPDATE CURRENT ADDRESS
 18041  107652  062737  000200  172354          ADD     #200,   KIPAR6
 18042  107660  022737  177600  172354          CMP     #177600,KIPAR6          ;REACHED I/O PAGE?
 18043  107666  001003                          BNE     10$                     ;BRANCH IF NOT
 18044  107670  162737  000200  172354          SUB     #200,KIPAR6             ;DON'T UPDATE PAR FOR I/O PAGE
 18045  107676  005237  003152          10$:    INC     ALLCTR                  ;IF ALL ADDRESSES ALLOCATED
 18046  107702  002737                          BLT     2$                      ;ENDDO
 18047  107704  052737  000004  177746          BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
 18048  107712  013737  003016  003020          MOV     LOWADD, GOODAD          ;INITIALIZE GOOD ADDRESS
 18049  107720  060137  003020                  ADD     R1,     GOODAD          ;
 18050  107724  012737  140000  111504          MOV     #140000,CURADD          ;GET FIRST ADDRESS
 18051  107732  060137  111504                  ADD     R1,     CURADD          ;START WITH 1 OR 2 LOCATION
 18052  107736  005037  172354                  CLR     KIPAR6
 18053  107742  072127  000006                  ASH     #6,R1                   ;
 18054  107746  060137  172354                  ADD     R1,KIPAR6               ;MAKE SURE ON THE RIGHT BOUNDARY
 18055  107752  012737  177600  003152          MOV     #-200,  ALLCTR          ;DO UNTIL ALL LOCATIONS CHECKED
 18056  107760  012737  177770  111460  5$:     MOV     #-10,   DCOUNT          ;DO UNTIL ADDRESS IN EACH BLOCK CHECKED
 18057  107766  013737  111504  001122  6$:     MOV     CURADD, $BDADR          ;IN CASE OF ERRORS
 18058  107774  042737  160000  001122          BIC     #160000,$BDADR          ;CLEAR PAR BITS
 18059  110002  042737  000004  177746          BIC     #BIT02, CCR             ;CLEAR FORCE MISS
 18060  110010  005777  001470                  TST     @CURADD                 ;READ CURRENT ADDRESS
 18061  110014  013737  177752  111464          MOV     HITMIS, RECDAT          ;STORE REGISTER
 18062  110022  032737  000004  111464          BIT     #BIT2,  RECDAT          ;IF ACCESS WAS A HIT
 18063  110030  001406                          BEQ     7$                      ;THEN
 18064  110032  023737  111504  003020          CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS NOT EQUAL TO GOOD
 18065  110040  001407                          BEQ     8$                      ;ADDRESS THEN
 18066  110042  104046                          ERROR   +46                     ;ERROR IN TAG STORE
 18067  110044  000405                          BR      8$                      ;
```

TEST - TAG STORE RAM TESTS

```
18068 110046  023737  111504  003020  7$:     CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS EQUAL GOOD ADDRESS
18069 110054  001001                          BNE     8$                      ;THEN
18070 110056  104047                          ERROR   +47                     ;ERROR IN TAG STORE
18071 110060  062737  001000  111504  8$:     ADD     #1000,  CURADD          ;UPDATE TO NEXT BLOCK
18072 110066  052737  000004  177746          BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
18073 110074  005237  111460                  INC     DCOUNT                  ;IF ADDRESS NOT CHECKED FOR EACH BLOCK
18074 110100  002732                          BLT     6$                      ;ENDDO
18075 110102  062737  000004  003020          ADD     #4,     GOODAD          ;UPDATE GOOD ADDRESS
18076 110110  162737  007774  111504          SUB     #7774,  CURADD          ;INCREMENT IN 2 WORDS
18077 110116  062737  000400  172354          ADD     #400,   KIPAR6
18078 110124  005237  003152                  INC     ALLCTR                  ;IF ALL ADDRESSES CHECKED
18079 110130  002713                          BLT     5$                      ;ENDDO
18080 110132  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
18081 110140  062701  000002                  ADD     #2,     R1              ;PREPARE FOR THE 2ND PASS THRU
18082 110144  022701  000002                  CMP     #2,     R1              ;DONE TWICE?
18083 110150  001002                          BNE     30$                     ;IF SO, EXIT
18084 110152  000137  107546                  JMP     9$
18085 110156  062737  002000  003016  30$:     ADD     #2000,  LOWADD          ;UPDATE TO NEXT BLOCK TO BE ALLOCATED
18086 110164  005237  003154                  INC     LOOPIN                  ;IF ALL BLOCKS WERE TESTED
18087 110170  002002                          BGE     15$                     ;
18088 110172  000137  107542                  JMP     1$                      ;ENDDO
18089 110176  012737  000400  177746  15$:     MOV     #400,   CCR             ;CLEAR DIAGNOSTIC BIT
18090 110204  005037  177572                  CLR     SR0                     ;DISABLE MMU
18091 110210  013737  001160  000004          MOV     $TMP0,#04               ;RESTORE TIMEOUT VECTOR
18092 110216  042737  000020  172516          BIC     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
18093 110224                          18$:
      110224  000401                          BR      TST25           ;;GO TO NEXT TEST
18094
18095 110226  000002                  20$:    RTI
18096
18097
```

```
18099                                   .SBTTL  TEST - STANDALONE MODE TEST
18100                                   ;THIS TEST VERIFIES THAT NMX CAN BE CREATED IN STANDALONE MODE.
18101                                   ;
18102                                   ;ALLOCATE INSTRUCTIONS IN CACHE
18103                                   ;GUARANTEE MISS ON TEST LOCATION
18104                                   ;IN STANALONE MODE ACCESS TEST LOCATION
18105                                   ;IF NO TIMEOUT THEN
18106                                   ;.      ERROR
18107                                   ;ENDIF
18108                                   ;
18109                                   ;;*********************************************************************
     110230  000004                     TST25:  SCOPE
18110 110232  000240                             NOP
18111 110234  005737  003032                     TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
18112 110240  001002                             BNE     99$                     ; NOT YET
18113 110242  000240                             NOP                             ; DEBUG AID
18114 110244  000452                             BR      TST26           ;;GO TO NEXT TEST
18115 110246  000240                     99$:    NOP
18116
18117 110250  012737  000400  177746             MOV     @400,CCR                ;FLUSH CACHE
18118 110256  012701  110256                     MOV     @.,R1                   ;START WITH CURRENT INSTRUCTION
18119 110262  005721                     1$:     TST     (R1)+                   ;READ A WORD
18120 110264  022701  110362                     CMP     @10$,R1                 ;DONE?
18121 110270  001374                             BNE     1$                      ;IF NOT, CONTINUE
18122 110272  005737  020000                     TST     @#20000                 ;TO GUARANTY MISS ON 0
18123 110276  013702  000004                     MOV     @#4,R2                  ;STORE TIMEOUT VECTOR
18124 110302  012737  110336  000004             MOV     @5$,@#4                 ;POINT NEW TO THE TEST
18125 110310  012737  000340  000006             MOV     @340,@#6                ;AT PRIORITY 7
18126 110316  042737  001000  177520             BIC     @#BIT09,@#BCSR          ;DISABLE HALT ON BREAK
18127 110324  052737  000400  177520             BIS     @#BIT08,@#BCSR          ;GO TO STANDALONE MODE
18128 110332  005737  000000                     TST     @#0                     ;MISS, SHOULD TIMEOUT
18129 110336  042737  000400  177520     5$:     BIC     @#BIT08,@#BCSR          ;CLEAR STANDALONE BIT
18130 110344  052737  001000  177520             BIS     @#BIT09,@#BCSR          ;ENABLE HALT ON BREAK
18131 110352  022716  110336                     CMP     @5$,(SP)                ;TIMEOUT?
18132 110356  001401                             BEQ     10$                     ;IF YES, BRANCH
18133 110360  104044                             ERROR   .44
18134 110362  012706  001100             10$:    MOV     @1100,SP                ;RESTORE STACK
18135 110366  010237  000004                     MOV     R2,@#4                  ;AND TIMEOUT VECTOR
```

```
18137                                   .SBTTL  TEST  - MOVING INVERSIONS TEST FOR DATA RAMS
18138                                   ;MOVING INVERSIONS TEST FOR DATA RAMS - THE TEST IS STARTED AFTER LOADING THE
18139                                   ;RAM STORE WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A
18140                                   ;1 IS SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE
18141                                   ;ADDRESS IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF
18142                                   ;THE WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED
18143                                   ;PLUGGING IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESSING IN THE DOWNWARD
18144                                   ;DIRECTION. FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE
18145                                   ;LSB. TO SAVE TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING
18146                                   ;ONLY A SINGLE BIT AT A TIME EVERY FOURTH BIT WILL BE DONE CONCURRENTLY.
18147                                   ;THIS TEST RUNS IN STANDALONE MODE.
18148                                   ;
18149                                   ;BGNTST
18150                                   ;SETUP AND ENABLE MMU
18151                                   ;SETUP CCR TO ABORT PARITY ERRORS
18152                                   ;CLEAR CACHE
18153                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
18154                                   ;LET FWDSEQ = #1
18155                                   ;LET ADDLSB = #1
18156                                   ;DO UNTIL ADDLSB EQ #20000
18157                                   ;.      LET CURDAT = 0
18158                                   ;.      LET RITEDA = #1
18159                                   ;.      LET NEWDAT = #1
18160                                   ;.      IF FWDSEQ = #1 THEN
18161                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18162                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18163                                   ;.      ELSE
18164                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18165                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18166                                   ;.      ENDIF
18167                                   ;.      LET CURADD = FSTADD
18168                                   ;.      LET DCOUNT = #0
18169                                   ;.      SAVE CONTENTS OF VECTOR 114
18170                                   ;.      LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
18171                                   ;.      DO UNTIL DCOUNT EQ #10
18172                                   ;.      .       LET RECDAT = @CURADD
18173                                   ;.      .       IF RECDAT NE CURDAT THEN
18174                                   ;.      .       .       LET R1 EQUAL CURRENT DATA
18175                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
18176                                   ;.      .       ENDIF
18177                                   ;.      .       IF DCOUNT GT #3 THEN
18178                                   ;.      .       .       LET @CURADD = @CURADD CLEARBY RITEDA
18179                                   ;.      .       ELSE
18180                                   ;.      .       .       LET @CURADD = @CURADD SETBY RITEDA
18181                                   ;.      .       ENDIF
18182                                   ;.      .       LET RECDAT = @CURADD
18183                                   ;.      .       IF RECDAT NE NEWDAT THEN
18184                                   ;.      .       .       LET R1 EQUAL NEW DATA
18185                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
18186                                   ;.      .       ENDIF
18187                                   ;.      .       IF CURADD EQ LASTAD THEN
18188                                   ;.      .       .       IF RITEDA = #210 THEN
18189                                   ;.      .       .       .       IF DCOUNT NE #7 THEN
18190                                   ;.      .       .       .       .       LET CURDAT = #377
18191                                   ;.      .       .       .       .       LET RITEDA = #1
18192                                   ;.      .       .       .       .       LET NEWDAT = #376
18193                                   ;.      .       .       .       ENDIF
```

```
18194                                   ;.      .       .       ELSE
18195                                   ;.      .       .       .       LET CURDAT = NEWDAT
18196                                   ;.      .       .       .       ROTATE RITEDA
18197                                   ;.      .       .       .       IF DCOUNT GT #3 THEN
18198                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT CLEAREDBY RITEDA
18199                                   ;.      .       .       .       ELSE
18200                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA
18201                                   ;.      .       .       .       ENDIF
18202                                   ;.      .       .       ENDIF
18203                                   ;.      .       .       INCREMENT DCOUNT
18204                                   ;.      .       .       LET CURADD = FSTADD
18205                                   ;.      .       ELSE
18206                                   ;.      .       .       IF FWDSEQ = #1 THEN
18207                                   ;.      .       .       .       LET CURADD = CURADD + ADDLSB
18208                                   ;.      .       .       .       IF CARRY THEN
18209                                   ;.      .       .       .       .       LET CURADD = CURADD + #1
18210                                   ;.      .       .       .       ENDIF
18211                                   ;.      .       .       ELSE
18212                                   ;.      .       .       .       LET CURADD = CURADD - ADDLSB
18213                                   ;.      .       .       .       IF CARRY THEN
18214                                   ;.      .       .       .       .       LET CURADD = LASTAD - ADDLSB
18215                                   ;.      .       .       .       .       LET CURADD = CURADD - #1
18216                                   ;.      .       .       .       ENDIF
18217                                   ;.      .       .       ENDIF
18218                                   ;.      .       ENDIF
18219                                   ;.      ENDDO
18220                                   ;.      IF FWDSEQ EQ #1 THEN
18221                                   ;.      .       LET FWDSEQ = #0
18222                                   ;.      ELSE
18223                                   ;.      .       ROTATE ADDLSB
18224                                   ;.      .       LET FWDSEQ = #1
18225                                   ;.      ENDIF
18226                                   ;ENDDO
18227                                   ;RESTORE VECTOR 114
18228                                   ;ENDTST
18229
18230                                   ;;**********************************************************************
    110372  000004                      TST26:  SCOPE
18231 110374  000240                          NOP
18232 110376  005737  003032                   TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
18233 110402  001003                          BNE     99$                     ; NOT YET
18234 110404  000240                          NOP                             ; DEBUG AID
18235 110406  000137  111506                  JMP     ENDMOV                  ; YES SKIP THIS
18236 110412  000240                    99$:  NOP
18237
18238 110414  032777  000200  070516         BIT     #BIT07,@SWR             ;RUN THIS TEST?
18239 110422  001002                          BNE     100$                    ;IF SET, GO DO IT
18240 110424  000137  111506                  JMP     ENDMOV                  ;OTHERWISE, GO TO NEXT TEST
18241 110430  042737  001000  177520  100$:  BIC     #1000,@CSR              ;DISABLE HALT ON BREAK
18242 110436  004737  134110                  JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
18243 110442  012737  002000  172354         MOV     #2000,KIPAR6            ;START ON 32K BOUNDARY
18244 110450  005237  177572                  INC     SR0                     ;TURN ON MMU
18245 110454  052737  000002  177746         BIS     #2,     CCR             ;SET DIAG. BIT
18246 110462  013737  000004  001160         MOV     #04,    $TMP0           ;SAVE 4
18247                                   ;
18248                                   ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
18249                                   ;
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18250 110470  012703  140000              MOV     #140000,R3          ;START FOR THE TEST
18251 110474  012704  150000              MOV     #150000,R4          ;LOWER BOUNDARY TEST AREA
18252 110500  012705  157777              MOV     #157777,R5          ;HIGH BOUNDARY TEST AREA
18253 110504  000406                      BR      2$
18254 110506  012703  150000        1$:   MOV     #150000,R3          ;START OF THE TEST
18255 110512  012704  140000              MOV     #140000,R4          ;LOWER BOUNDARY TEST AREA
18256 110516  012705  147777              MOV     #147777,R5          ;HIGH BOUNDARY
18257 110522  012702  110642        2$:   MOV     #STMOVI,R2          ;START WITH CURRENT
18258 110526  010300                      MOV     R3,R0               ;MOVE TO UPPER 4K
18259 110530  012220                3$:   MOV     (R2)+,(R0)+         ;WORD BY WORD
18260 110532  022702  111506              CMP     #ENDMOV,R2          ;ALL DONE
18261 110536  001374                      BNE     3$                  ;
18262 110540  010400                      MOV     R4,R0               ;CLEAR CACHE UNDER TEST
18263 110542  012701  004000              MOV     #4000.  R1
18264 110546  005020                4$:   CLR     (R0)+
18265 110550  077102                      SOB     R1.     4$
18266 110552  004713                      JSR     PC,(R3)             ;GO DO THE ROUTINE
18267 110554  005702                      TST     R2                  ;ANY ERRORS?
18268 110556  001411                      BEQ     5$                  ;IF NOT, CONTINUE
18269 110560  010037  111464              MOV     R0,RECDAT           ;DATA RECEIVED
18270 110564  010237  001122              MOV     R2,$BDADR           ;ADDRESS RECIEVED
18271 110570  042737  140000  001122      BIC     #140000,$BDADR      ;STRIP OF PAR BITS
18272 110576  104043                      ERROR   +43                 ;
18273 110600  000403                      BR      6$                  ;EXIT
18274 110602  022703  150000        5$:   CMP     #150000,R3          ;DONE FOR BOTH HALVES?
18275 110606  001337                      BNE     1$                  ;IF NOT, DO AGAIN
18276 110610  052737  001000  177520  6$:   BIS     #BIT09,BCSR         ;ENABLE HALT ON BREAK
18277 110616  013737  001160  000004      MOV     $TMP0,@#4           ;RESTORE TIMEOUT VECTOR
18278 110624  012737  000400  177746      MOV     #400,CCR            ;INIT CCR FOR EXIT
18279 110632  005037  177572              CLR     SR0                 ;TURN OFF MMU
18280 110636  000137  111506              JMP     ENDMOV              ;GO TO NEXT TEST
18281
18282                             .DSABL AMA
18283 110642  052737  000400  177520  STMOVI: BIS     #BIT08,@#BCSR       ;STANDALONE MODE
18284 110650  005002                      CLR     R2                  ;ERROR INDICATOR
18285 110652  012767  000001  000606      MOV     #1.     FWDSEQ      ;INIT UPWARD ADDRESSING INDICATOR
18286 110660  012767  000001  000602      MOV     #1.     ADDLSB      ;INIT LSB AND DO LOOP UNTIL SHIFTED OUT
18287 110666  042737  100000  172300      BIC     #100000,@#KIPDR0    ;NO BYPASS
18288 110674  012737  172360  000004      MOV     #KDPAR0,@#4         ;ALLOCATE TIMEOUT VECTOR
18289 110702  012737  000340  000006      MOV     #340.   @#6         ;AT PRIORITY 7
18290 110710  012737  000006  172360      MOV     #6.     @#KDPAR0    ;PUT RETURN
18291 110716  052737  100000  172300      BIS     #100000,@#KIPDR0    ;BYPASS
18292 110724  005067  000546        TSTLUP: CLR     CURDAT              ;INIT CURRENT DATA
18293 110730  012767  000021  000534      MOV     #21.    RITEDA      ;INIT DATA TO BE WRITTEN
18294 110736  012767  000021  000530      MOV     #21.    NEWDAT      ;INIT EXPECTED DATA
18295 110744  005767  000516              TST     FWDSEQ              ;IF ADDRESSING UPWARD
18296 110750  001405                      BEQ     1$                  ;THEN
18297 110752  010467  000522              MOV     R4,FSTADD           ;LET FIRST ADDRESS EQUAL LOWEST VALUE
18298 110756  010567  000520              MOV     R5,LSTADD           ;LET LAST ADDRESS EQUAL HIGHEST VALUE
18299 110762  000404                      BR      2$                  ;ELSE
18300 110764  010567  000510        1$:   MOV     R5,FSTADD           ;LET FIRST ADDRESS EQUAL HIGHEST VALUE
18301 110770  010467  000506              MOV     R4,LSTADD           ;LET LAST ADDRESS EQUAL LOWEST VALUE
18302 110774  016767  000500  000502  2$:   MOV     FSTADD, CURADD      ;LET CURRENT ADDRESS EQUAL FIRST ADDRESS
18303 111002  005067  000452              CLR     DCOUNT              ;INIT LOOP COUNTER
18304 111006  022767  000010  000444  BGNTLP: CMP     #10.    DCOUNT      ;DO LOOP 8 TIMES (4 TIMES TO WRITE 1'S
18305 111014  001567                      BEQ     ENDTLP              ;AND 4 TIMES TO WRITE BACK 0'S)
18306                                     ;
```

```
18307                                   ; DON'T REWRITE TIMEOUT VECTOR
18308                                   ;
18309 111016  016700  000462                    MOV     CURADD, R0          ;STORE CURRENT ADDRESS
18310 111022  042700  170000                    BIC     #170000,R0          ;LEAVE ONLY LOW 4K BITS
18311 111026  022700  000004                    CMP     #4,     R0          ;TIMOUT VECTOR?
18312 111032  001531                            BEQ     8$                  ;IF SO, DON'T REWRITE IT
18313 111034  022700  000005                    CMP     #5,     R0          ;
18314 111040  001526                            BEQ     8$                  ;
18315 111042  022700  000006                    CMP     #6,     R0          ;
18316 111046  001523                            BEQ     8$                  ;
18317 111050  022700  000007                    CMP     #7,     R0          ;
18318 111054  001520                            BEQ     8$                  ;
18319                                   ;
18320                                   ;CHECK PATTERNS NOW
18321                                   ;
18322 111056  016701  000414                    MOV     CURDAT, R1          ;MOVE GOOD DATA TO R1
18323 111062  117767  000416  000374            MOVB    @CURADD,RECDAT      ;FIRST READ LOCATION
18324 111070  126767  000370  000400            CMPB    RECDAT, CURDAT      ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18325 111076  001402                            BEQ     1$                  ;DATA THEN
18326 111100  000167  000334                    JMP     EXBAD               ;EXIT TEST
18327 111104  016701  000364            1$:     MOV     NEWDAT, R1          ;MOVE GOOD DATA TO R1
18328 111110  022767  000003  000342            CMP     #3,     DCOUNT      ;IF LOOP COUNT GREATER THAN 3
18329 111116  002004                            BGE     2$                  ;THEN
18330 111120  146777  000346  000356            BICB    RITEDA, @CURADD     ;CLEAR TEST DATA BY WRITE DATA
18331 111126  000403                            BR      3$                  ;ELSE
18332 111130  156777  000336  000346    2$:     BISB    RITEDA, @CURADD     ;SET TEST DATA BY WRITE DATA
18333 111136  117767  000342  000320    3$:     MOVB    @CURADD,RECDAT      ;DO READ AFTER WRITE
18334 111144  126767  000314  000322            CMPB    RECDAT, NEWDAT      ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18335 111152  001402                            BEQ     4$                  ;DATA THEN
18336 111154  000167  000260                    JMP     EXBAD               ;EXIT TEST
18337 111160  026767  000320  000314    4$:     CMP     CURADD, LSTADD      ;IF CURRENT ADDRESS EQUALS LAST ADDRESS
18338 111166  001053                            BNE     8$                  ;THEN
18339 111170  022767  000210  000274            CMP     #210,   RITEDA      ;AND IF WRITE PATTERN EQUALS LAST
18340 111176  001016                            BNE     5$                  ;PATTERN THEN
18341 111200  022767  000007  000252            CMP     #7,     DCOUNT      ;AND TEST IS NOT ON LAST LOOP
18342 111206  001435                            BEQ     7$                  ;THEN
18343 111210  012767  000377  000260            MOV     #377,   CURDAT      ;SET UP TO WRITE 0'S
18344 111216  012767  000021  000246            MOV     #21,    RITEDA
18345 111224  012767  000356  000242            MOV     #356,   NEWDAT
18346 111232  000423                            BR      7$                  ;ELSE
18347 111234  016767  000234  000234    5$:     MOV     NEWDAT, CURDAT      ;SET UP FOR NEXT DATA PATTERN
18348 111242  000241                            CLC
18349 111244  106167  000222                    ROLB    RITEDA
18350 111250  005567  000216                    ADC     RITEDA              ;GET A PATTERN
18351 111254  022767  000003  000176            CMP     #3,     DCOUNT      ;AND IF DOWNWARD ADDRESSING
18352 111262  002004                            BGE     6$                  ;THEN
18353 111264  046767  000202  000202            BIC     RITEDA, NEWDAT      ;LET NEWDAT BE CLEARED BY WRITE DATA
18354 111272  000403                            BR      7$                  ;ELSE
18355 111274  056767  000172  000172    6$:     BIS     RITEDA, NEWDAT      ;LET NEWDAT BE SET BY WRITE DATA
18356 111302  005267  000152            7$:     INC     DCOUNT              ;UPDATE LOOP COUNTER
18357 111306  016767  000166  000170            MOV     FSTADD, CURADD      ;REINIT FIRST ADDRESS FOR THIS PASS
18358 111314  000426                            BR      10$                 ;ELSE
18359 111316  005767  000144            8$:     TST     FWDSEQ              ;IF ADDRESSING UPWARD
18360 111322  001412                            BEQ     9$                  ;THEN
18361 111324  066767  000140  000152            ADD     ADDLSB, CURADD      ;CALCULATE NEXT HIGHER ADDRESS
18362 111332  020567  000146                    CMP     R5,CURADD           ;IF CURRENT ADDRESS HAS BEEN INCREASED
18363 111336  002015                            BGE     10$                 ;ABOVE HIGHEST ADDRESS THEN
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18364 111340  162767  007777  000136          SUB     #7777,  CURADD          ;ROLL ADDRESS BACK
18365 111346  000411                          BR      10$                     ;ELSE
18366 111350  166767  000114  000126  9$:     SUB     ADDLSB, CURADD          ;CALCULATE NEXT LOWER ADDRESS
18367 111356  020467  000122                  CMP     R4,CURADD               ;IF CURRENT ADDRESS HAS BEEN DECREASED
18368 111362  003403                          BLE     10$                     ;BELOW LOWEST ADDRESS THEN
18369 111364  062767  007777  000112          ADD     #7777,  CURADD          ;ROLL ADDRESS BACK
18370 111372  000605                  10$:    BR      BGNTLP                  ;ENDDO
18371 111374  005767  000066          ENDTLP: TST     FWDSEQ                  ;IF ADDRESSING UPWARD FINISHED
18372 111400  001404                          BEQ     1$                      ;THEN
18373 111402  005067  000060                  CLR     FWDSEQ                  ;DO ADDRESSING DOWNWARD
18374 111406  000167  177312                  JMP     TSTLUP
18375 111412  012767  000001  000046  1$:     MOV     #1,     FWDSEQ          ;SET ADDRESSING UPWARD INDICATOR
18376 111420  006167  000044                  ROL     ADDLSB                  ;UPDATE LSB TO NEXT POSITION
18377 111424  022767  020000  000036  ENDLUP: CMP     #20000,ADDLSB           ;ALL DONE?
18378 111432  001406                          BEQ     EXITST                  ;ENDDO
18379 111434  000167  177264                  JMP     TSTLUP
18380 111440  016700  000020          EXBAD:  MOV     RECDAT,R0               ;STORE RECEIVED DATA
18381 111444  016702  000034                  MOV     CURADD,R2               ;STORE ADDRESS
18382 111450  042737  000400  177520  EXITST: BIC     #BIT08,@#CSR            ;OUT OF STANDALONE
18383 111456  000207                          RTS     PC
18384                                 .ENABL AMA
18385
18386 111460  000000                  DCOUNT: .WORD   0
18387 111462  000000                  EXPDAT: .WORD   0               ;STORES EXPECTED (GOOD) DATA FOR COMPARISONS
18388 111464  000000                  RECDAT: .WORD   0               ;STORES RECIEVED DATA TO BE VERIFIED
18389 111466  000000                  FWDSEQ: .WORD   0               ;USED TO INDICATE DIRECTION OF ADDRESSING
18390 111470  000000                  ADDLSB: .WORD   0               ;STORES LEAST SIGNIFICANT BIT FOR RAM TESTS
18391 111472  000000                  RITEDA: .WORD   0               ;STORES WRITE DATA FOR RAM TESTS
18392 111474  000000                  NEWDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18393 111476  000000                  CURDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18394 111500  000000                  FSTADD: .WORD   0               ;STORES FIRST ADDRESS IN ADDRESSING SEQUENCE
18395 111502  000000                  LSTADD: .WORD   0               ;STORES LAST ADDRESS IN ADDRESSING SEQUENCE
18396 111504  000000                  CURADD: .WORD   0               ;STORES CURRENT ADDRESS FOR RAM TESTS
18397
18398 111506                          ENDMOV:
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18400                                   .SBTTL  TEST  - MOVING INVERSIONS TEST FOR TAG STORE
18401                                   ;MOVING INVERSIONS TEST - THE TEST IS STARTED AFTER LOADING THE RAM STORE
18402                                   ;WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A 1 IS
18403                                   ;SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE ADDRESS
18404                                   ;IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF THE
18405                                   ;WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED PLUGGING
18406                                   ;IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESS IN THE DOWARD DIRECTION.
18407                                   ;FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE LSB. TO SAVE
18408                                   ;TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING ONLY A SINGLE
18409                                   ;BIT AT A TIME EVERY THIRD BIT WILL BE DONE CONCURRENTLY. ALSO NOTE THAT
18410                                   ;SINCE THE TAG STORE CANNOT BE DIRECTLY ACCESSED THE ENTIRE PATTERN MUST BE
18411                                   ;DONE BY DOING MEMORY CYCLES TO THE CORRECT BUS ADDRESSES. TO DO THIS THE
18412                                   ;TEST WILL BE DONE WITH THE DIAGNOSTIC BIT IN THE CCR (BIT 1) SET TO A 1.
18413                                   ;THIS TEST RUNS IN STANDALONE MODE.
18414                                   ;
18415                                   ;
18416                                   ;BGNTST
18417                                   ;SETUP AND ENABLE MMU
18418                                   ;SETUP CCR TO ABORT PARITY ERRORS
18419                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
18420                                   ;LET FWDSEQ = #1
18421                                   ;LET ADDLSB = #1
18422                                   ;DO UNTIL ADDLSB EQ #20000
18423                                   ;.      LET NEWDAT = 22200
18424                                   ;.      LET CURDAT = 0
18425                                   ;.      IF FWDSEQ = #1 THEN
18426                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18427                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18428                                   ;.      ELSE
18429                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18430                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18431                                   ;.      ENDIF
18432                                   ;.      LET DCOUNT = #0
18433                                   ;.      DO UNTIL DCOUNT EQ #10
18434                                   ;.      .       READ CURADD USING CURDAT AS PAR
18435                                   ;.      .       IF MISS THEN
18436                                   ;.      .       .       ERROR
18437                                   ;.      .       ENDIF
18438                                   ;.      .       WRITE CURADD USING NEWDAT AS PAR
18439                                   ;.      .       ENDIF
18440                                   ;.      .       IF HIT THEN
18441                                   ;.      .       .       ERROR
18442                                   ;.      .       ENDIF
18443                                   ;.      .       READ CURADD USING NEWDAT AS PAR
18444                                   ;.      .       IF MISS THEN
18445                                   ;.      .       .       ERROR
18446                                   ;.      .       ENDIF
18447                                   ;.      .       IF CURADD EQ LASTAD THEN
18448                                   ;.      .       .       LET CURDAT = NEWDAT
18449                                   ;.      .       .       IF DCOUNT < #1 THEN
18450                                   ;.      .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA ROTATED LEFT
18451                                   ;.      .       .       ENDIF
18452                                   ;.      .       .       IF DCOUNT > #1 THEN
18453                                   ;.      .       .       .       LET NEWDAT = NEWDAT CLR BY RITEDA ROTATED LEFT
18454                                   ;.      .       .       ELSE
18455                                   ;.      .       .       .       LET NEWDAT = #155400 (NO ALL 1'S)
19456                                   ;.      .       .       .       LET RITEDA = #22200
```

```
18457                                   ;.      .       .       ENDIF
18458                                   ;.      .       .       INCREMENT DCOUNT
18459                                   ;.      .       ELSE
18460                                   ;.      .       .       IF FWDSEQ = #1 THEN
18461                                   ;.      .       .       .       LET CURADD = CURADD + ADDLSB
18462                                   ;.      .       .       .       IF CURADD GE HIGH ADDRESS THEN
18463                                   ;.      .       .       .       .       LET CURADD = CURADD + #7776
18464                                   ;.      .       .       .       ENDIF
18465                                   ;.      .       .       ELSE
18466                                   ;.      .       .       .       LET CURADD = CURADD - ADDLSB
18467                                   ;.      .       .       .       IF CURADD LE LOW ADDRESS THEN
18468                                   ;.      .       .       .       .       LET CURADD = CURADD - #7776
18469                                   ;.      .       .       .       ENDIF
18470                                   ;.      .       .       ENDIF
18471                                   ;.      .       ENDIF
18472                                   ;.      ENDDO
18473                                   ;.      IF FWDSEQ EQ #1 THEN
18474                                   ;.      .       LET FWDSEQ = #0
18475                                   ;.      ELSE
18476                                   ;.      .       ROTATE ADDLSB
18477                                   ;.      .       LET FWDSEQ = #1
18478                                   ;.      ENDIF
18479                                   ;ENDDO
18480                                   ;RESTORE PARITY ABORT VECTOR
18481                                   ;ENDTST
18482
18483                                   ;;*****************************************************************
     111506  000004                   TST27:  SCOPE
18484 111510  000240                           NOP
18485 111512  005737  003032                    TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
18486 111516  001003                           BNE     99$                     ; NOT YET
18487 111520  000240                           NOP                             ; DEBUG AID
18488 111522  000137  112630                   JMP     ENDTAG                  ; YES SKIP THIS
18489 111526  000240                   99$:    NOP
18490
18491 111530  032777  000100  067402           BIT     #BIT06,@SWR             ;RUN THIS TEST?
18492 111536  001002                           BNE     100$                    ;IF SET, GO DO IT
18493 111540  000137  112630                   JMP     ENDTAG                  ;OTHERWISE, GO TO NEXT TEST
18494 111544  042737  001000  177520   100$:   BIC     #1000,@CSR              ;DISABLE HALT ON BREAK
18495 111552  004737  134110                   JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
18496 111556  005237  177572                   INC     @SR0                    ;TURN ON MMU
18497 111562  012737  000020  172516           MOV     #BIT04,@MMR3            ;ENABLE 22-BIT MAPPING
18498 111570  052737  000002  177746           BIS     #2,     @CCR            ;SET DIAG. BIT
18499 111576  013737  000004  001160           MOV     @#4,    $TMPO           ;STORE TIMEOUT VECTOR
18500                                   ;
18501                                   ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
18502                                   ;
18503 111604  012703  140000                   MOV     #140000,R3              ;START FOR THE TEST
18504 111610  012704  130000                   MOV     #130000,R4              ;LOWER BOUNDARY TEST AREA
18505 111614  012705  137776                   MOV     #137776,R5              ;HIGH BOUNDARY TEST AREA
18506 111620  000406                           BR      2$
18507 111622  012703  150000           1$:     MOV     #150000,R3              ;START OF THE TEST
18508 111626  012704  120000                   MOV     #120000,R4              ;LOWER BOUNDARY TEST AREA
18509 111632  012705  127776                   MOV     #127776,R5              ;HIGH BOUNDARY
18510 111636  012737  002000  172354   2$:     MOV     #2000,@KIPAR6           ;ABOVE 32K
18511 111644  012702  111450                   MOV     #EXITST,R2              ;START WITH CURRENT
18512 111650  010300                           MOV     R3,R0                   ;MOVE TO UPPER 4K
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18513 111652  012220                        3$:     MOV     (R2)+,(R0)+             ;WORD BY WORD
18514 111654  022702  112630                        CMP     @ENDTAG,R2              ;ALL DONE?
18515 111660  001374                                BNE     3$                      ;
18516 111662  012700  111764                        MOV     #STMOVT,R0              ;ADDRESS OF ROUTINE
18517 111666  162700  111450                        SUB     #EXITST,R0              ;PROPER OFSET
18518 111672  050003                                BIS     R0,R3                   ;
18519 111674  004713                                JSR     PC,(R3)                 ;GO DO THE ROUTINE
18520 111676  005700                                TST     R0                      ;ANY ERRORS?
18521 111700  001407                                BEQ     4$                      ;CONTINUE
18522 111702  010037  001122                        MOV     R0,$BDADR               ;STORE FAILED ADDRESS
18523 111706  042737  160000  001122                BIC     #160000,$BDADR          ;CLEAR PAR
18524 111714  104050                                ERROR   .50
18525 111716  000403                                BR      5$                      ;EXIT TEST
18526 111720  022703  150000              4$:     CMP     #150000,R3              ;DONE FOR BOTH HALVES?
18527 111724  103736                                BLO     1$                      ;IF NOT, DO AGAIN
18528 111726  013737  001160  000004    5$:     MOV     $TMP0,  @#4             ;RESTORE TIMEOUT VECTOR
18529 111734  012737  000400  177746                MOV     #400,   CCR             ;INIT CCR FOR EXIT
18530 111742  005037  177572                        CLR     SR0                     ;TURN OFF MMU
18531 111746  005037  172516                        CLR     MMR3                    ;
18532 111752  052737  001000  177520                BIS     #1000,@BCSR             ;ENABLE HALT ON BREAK
18533 111760  000137  112630                        JMP     ENDTAG                  ;EXIT TEST
18534
18535                                       .DSABL  AMA
18536 111764  052737  000400  177520    STMOVT: BIS     #BIT08,@#BCSR           ;STANDALONE MODE
18537 111772  042737  100000  172312                BIC     #BIT15, @#KIPDR5        ;NO BYPASS
18538 112000  042737  100000  172300                BIC     #100000,@#KIPDR0        ;NO BYPASS
18539 112006  012737  172360  000004                MOV     #KDPAR0,@#4             ;ALLOCATE TIMEOUT VECTOR
18540 112014  012737  000340  000006                MOV     #340,   @#6             ;AT 7
18541 112022  012737  000006  172360                MOV     #6,     @#KDPAR0        ;PUT RETURN
18542 112030  052737  100000  172300                BIS     #100000,@#KIPDR0        ;BYPASS
18543 112036  005037  172352                        CLR     @#KIPAR5                ;
18544 112042  010400                                MOV     R4,R0                   ;CLEAR CACHE UNDER TEST
18545 112044  012701  004000                        MOV     #4000,  R1              ;
18546 112050  005020                        4$:     CLR     (R0)+                   ;
18547 112052  077102                                SOB     R1,     4$              ;
18548 112054  005000                                CLR     R0                      ;CLEAR ERROR FLAG
18549 112056  012767  000001  177402                MOV     #1,     FWDSEQ          ;SET UPWARD ADDRESSING INDICATOR
18550 112064  012767  000002  177376                MOV     #2,     ADDLSB          ;INITIALIZE LEAST SIGNIFIGANT BIT
18551 112072  005067  177400              TSLOOP: CLR     CURDAT                  ;OLD DATA
18552 112076  012767  022200  177370                MOV     #22200,NEWDAT           ;SET ADDRESS BITS
18553 112104  012767  022200  177360                MOV     #22200,RITEDA           ;SET ADDRESS BITS
18554 112112  005767  177350                        TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18555 112116  001405                                BEQ     1$                      ;THEN
18556 112120  010467  177354                        MOV     R4,     FSTADD          ;FIRST ADDRESS WILL BE LOWEST ADDRESS
18557 112124  010567  177352                        MOV     R5,     LSTADD          ;AND LAST ADDRESS WILL BE HIGHEST
18558 112130  000404                                BR      2$                      ;ELSE
18559 112132  010567  177342              1$:     MOV     R5,     FSTADD          ;FIRST ADDRESS WILL BE HIGHEST ADDRESS
18560 112136  010467  177340                        MOV     R4,     LSTADD          ;AND LAST ADDRESS WILL BE LOWEST
18561 112142  005067  177312              2$:     CLR     DCOUNT                  ;INITIALIZE LOOP COUNTER
18562 112146  016767  177326  177330                MOV     FSTADD, CURADD
18563                                               ;
18564                                       ; DON'T REWRITE TIMEOUT VECTOR
18565                                               ;
18566 112154  016700  177324              3$:     MOV     CURADD, R0              ;STORE CURRENT ADDRESS
18567 112160  042700  170000                        BIC     #170000,R0              ;LEAVE ONLY LOW 4K BITS
18568 112164  022700  000004                        CMP     #4,     R0              ;TIMOUT VECTOR?
18569 112170  001526                                BEQ     16$                     ;IF SO, DON'T REWRITE IT
```

```
18570 112172  022700  000006                CMP     #6,     R0              ;OR PRIORITY
18571 112176  001523                        BEQ     16$                     ;
18572 112200  016737  177272  172352        MOV     CURDAT, @#KIPAR5        ;GET OLD PATTERN
18573 112206  005777  177272                TST     @CURADD                 ;OLD DATA OK?
18574 112212  013767  177752  177244        MOV     @#HITMIS,RECDAT
18575 112220  032767  000004  177236  5$:   BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
18576 112226  001002                        BNE     6$                      ;THEN
18577 112230  000167  000346                JMP     EXBAD2                  ;ERROR, EXIT
18578 112234  016737  177234  172352  6$:   MOV     NEWDAT, @#KIPAR5        ;GET NEW PATTERN
18579 112242  005077  177236                CLR     @CURADD                 ;REGISTER AND WRITE LOCATION
18580 112246  013767  177752  177210        MOV     @#HITMIS,RECDAT
18581 112254  032767  000004  177202  8$:   BIT     #BIT02, RECDAT          ;IF ACCESS WAS A HIT
18582 112262  001406                        BEQ     9$                      ;THEN
18583 112264  032767  000010  177172  10$:  BIT     #BIT03, RECDAT          ;MISS?
18584 112272  001402                        BEQ     9$                      ;IF SO, CONTINUE
18585 112274  000167  000302                JMP     EXBAD2                  ;ERROR
18586 112300  005777  177200          9$:   TST     @CURADD                 ;REGISTER AND READ LOCATION
18587 112304  013767  177752  177152        MOV     @#HITMIS,RECDAT
18588 112312  032767  000004  177144  11$:  BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
18589 112320  001002                        BNE     12$                     ;THEN
18590 112322  000167  000254                JMP     EXBAD2                  ;ERROR, EXIT
18591 112326  026767  177150  177150  12$:  CMP     LSTADD, CURADD          ;IF CURRENT ADDRESS IS LAST ADDRESS
18592 112334  001044                        BNE     16$                     ;THEN
18593 112336  016767  177132  177132        MOV     NEWDAT,CURDAT           ;NEW KIPAR5
18594 112344  022767  000001  177106        CMP     #1,     DCOUNT          ;IF LOOP COUNTER LESS THAN 1
18595 112352  003406                        BLE     13$                     ;THEN
18596 112354  006367  177112                ASL     RITEDA                  ;UPDATE OF NEW DATA....
18597 112360  056767  177106  177106        BIS     RITEDA, NEWDAT          ;BY SETTING BITS
18598 112366  000421                        BR      15$                     ;ENDIF
18599 112370  022767  000001  177062  13$:  CMP     #1,     DCOUNT          ;CHANGE PATTERN?
18600 112376  001007                        BNE     14$                     ;IF SO, BRANCH
18601 112400  012767  022200  177064        MOV     #22200, RITEDA          ;START WITH THE SAME
18602 112406  012767  155400  177060        MOV     #155400,NEWDAT          ;DON'T DO ALL 1'S
18603 112414  000406                        BR      15$                     ;
18604 112416  006367  177050          14$:  ASL     RITEDA                  ;UPDATE OF NEW DATA....
18605 112422  046767  177044  177044        BIC     RITEDA, NEWDAT          ;BY CLEARING BITS
18606 112430  000400                        BR      15$                     ;ELSE
18607 112432  005267  177022          15$:  INC     DCOUNT                  ;INCREMENT THE LOOP COUNTER
18608 112436  016767  177036  177040        MOV     FSTADD, CURADD          ;FINISH FIRST CURRENT ADDRESS
18609 112444  000426                        BR      18$                     ;ELSE
18610 112446  005767  177014          16$:  TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18611 112452  001412                        BEQ     17$                     ;THEN
18612 112454  066767  177010  177022        ADD     ADDLSB, CURADD          ;ADD THE VALUE OF THE CURRENT LSB
18613 112462  020567  177016                CMP     R5,     CURADD          ;IF CURRENT ADDRESS GREATER THAN HIGHEST
18614 112466  002015                        BGE     18$                     ;VIRTUAL ADDRESS THEN
18615 112470  162767  007776  177006        SUB     #7776,  CURADD          ;ROLL IT BACK
18616 112476  000411                        BR      18$                     ;ELSE
18617 112500  166767  176764  176776  17$:  SUB     ADDLSB, CURADD          ;IF ADDRESSING DOWNWARD THEN SUBTRACT LSB
18618 112506  020467  176772                CMP     R4,     CURADD          ;IF CURRENT ADDRESS LESS THEN LOWEST
18619 112512  003403                        BLE     18$                     ;VIRTUAL ADDRESS THEN
18620 112514  062767  007776  176762        ADD     #7776,  CURADD          ;ROLL IT BACK
18621 112522  022767  000005  176730  18$:  CMP     #5,     DCOUNT          ;IF LOOP COUNTER LESS THAN 7
18622 112530  003402                        BLE     181$                    ;THEN LOOP BACK FOR NEXT BIT POSITION
18623 112532  000167  177416                JMP     3$
18624 112536  005767  176724          181$: TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18625 112542  001404                        BEQ     19$                     ;THEN
18626 112544  005067  176716                CLR     FWDSEQ                  ;START ADDRESSING DOWNWARD
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18627 112550  000167  177316                    JMP     TSLOOP                  ;ELSE
18628 112554  012767  000001  176704  19$:      MOV     #1,     FWDSEQ          ;START ADDRESSING UPWARD
18629 112562  006167  176702                    ROL     ADDLSB                  ;ROTATE LSB TO NEXT POSITION
18630 112566  022767  020000  176674            CMP     #20000, ADDLSB          ;ALL DONE?
18631 112574  001406                            BEQ     TSEND                   ;EXIT
18632 112576  000167  177270                    JMP     TSLOOP                  ;ENDDO
18633 112602  013701  172352          EXBAD2:   MOV     @#KIPAR5,R1             ;STORE PAR
18634 112606  016700  176672                    MOV     CURADD, R0              ;AND ADDRESS
18635 112612  012737  001200  172352  TSEND:    MOV     #1200,  @#KIPAR5        ;RESTORE PAR
18636 112620  042737  000400  177520            BIC     #BIT08, @#BCSR          ;OUT OF STANDALONE MODE
18637 112626  000207                            RTS     PC                      ;RETURN
18638 112630                          ENDTAG:
18639                                 .ENABL AMA
18640
```

TEST - PCR READ/WRITE BITS

```
                                .SBTTL  TEST  - PCR READ/WRITE BITS
                                ;PCR AND BCSR READ/WRITE BITS
                                ;THE FIRST TEST WILL CHECK THAT PCR REGISTER IS BOTH WORD AND
                                ;BYTE ADDRESSABLE. BITS 14-09 AND 06-01 WILL BE WRITTEN AND READ
                                ;AS ZEROES AND ONES. THE REST OF THE BITS HAVE TO BE ALL 0'S.
                                ;ROUTINE TEST
                                ;.      SAVE PCR
                                ;.      LET PCR=0
                                ;.      DO FOR PATTERN=001111,110011,101010,010101
                                ;.      .       WRITE PCR<14-09>=PATTERN
                                ;.      .       WRITE PCR<06-01>=PATTERN
                                ;.      .       IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PREVIOUS
                                ;.      .                                               PATTERN
                                ;.      .       .       THEN ERROR
                                ;.      .       ENDIF
                                ;.      .       WRITE PCR<06-01>=PATTERN
                                ;.      .       IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PATTERN
                                ;.      .       .       THEN ERROR
                                ;.      .       ENDIF
                                ;.      ENDDO
                                ;.      WRITE PCR=0101010101010101
                                ;.      IF PCR NE 0101010001010100 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;ENDROUTINE

                                ;;*****************************************************************
   112630  000004               TST30:  SCOPE
18669 112632  005037  177522          CLR     PCR                     ;INITIALIZE PCR TO 0
18670 112636  012702  112775          MOV     #SIXBIT+1,R2            ;R2->TABLE OF PATTERNS FOR 6 R/W BITS IN EAC
H BYTE
18671 112642  012704  112774          MOV     #SIXBIT,R4              ;R3 POINTER TO PREVIOUS PATTERN
18672 112646  012703  000004          MOV     #4,R3                   ;DO 4 TIMES
18673                                 ;
18674                                 ; WRITE TO HIGH BYTE FIRST
18675                                 ;
18676 112652  111237  177523  1$:     MOVB    (R2),PCR+1              ;WRITE TO HIGH BYTE
18677 112656  121237  177523          CMPB    (R2),PCR+1              ;BYTE WRITTEN OK?
18678 112662  001003                 BNE     2$                      ;IF NOT, BRANCH
18679 112664  121437  177522          CMPB    (R4),PCR                ;LOW BYTE CHANGED?
18680 112670  001405                 BEQ     3$                      ;IF NOT, BRANCH
18681 112672  111237  001125  2$:     MOVB    (R2),$GDDAT+1           ;EXPECTED PATTERN HIGH BYTE
18682 112676  111437  001124          MOVB    (R4),$GDDAT             ;LOW BYTE
18683 112702  104051                 ERROR   +51                     ;ERROR PCR READ/WRITE BITS
18684 112704  105724          3$:     TSTB    (R4)+                   ;INCREMENT POINTER FOR OLD
18685                                 ;
18686                                 ; WRITE TO LOW BYTE
18687                                 ;
18688 112706  111237  177522          MOVB    (R2),PCR                ;WRITE TO LOW BYTE
18689 112712  121237  177522          CMPB    (R2),PCR                ;BYTE WRITTEN OK?
18690 112716  001003                 BNE     4$                      ;IF NOT,BRANCH
18691 112720  121237  177522          CMPB    (R2),PCR                ;HIGH BYTE CHANGED?
18692 112724  001405                 BEQ     5$                      ;IF NOT, BRANCH
18693 112726  111237  001124  4$:     MOVB    (R2),$GDDAT             ;EXPECTED PATTERN HIGH BYTE
18694 112732  111237  001124          MOVB    (R2),$GDDAT             ;LOW BYTE
18695 112736  104051                 ERROR   +51                     ;ERROR PCR READ/WRITE BITS
18696 112740  105722          5$:     TSTB    (R2)+                   ;INCREMENT POINTER FOR NEW PATTERN
18697 112742  077335                 SOB     R3,1$                   ;DO FOR ALL 4 PATTERNS
```

TEST - PCR READ/WRITE BITS

```
18698                                   ;
18699                                   ; NOW TRY WORD ADDRESSING
18700                                   ;
18701 112744  012737  052525  177522          MOV     #52525,PCR                      ;WRITE A PATERN
18702 112752  022737  052124  177522          CMP     #52124,PCR                      ;ALL BUT BITS <8,7,0> OK?
18703 112760  001404                          BEQ     6$                              ;IF SO, BRANCH
18704 112762  112737  052124  001124          MOVB    #52124,$GDDAT                   ;EXPECTED PATTERN
18705 112770  104051                          ERROR   +51                             ;ERROR PCR READ/WRITE BITS
18706 112772                            6$:
      112772  000403                          BR      TST31           ;;GO TO NEXT TEST
18707
18708 112774     000     036     146  SIXBIT: .BYTE 0,36,146,124,52                   ;001111,110011,101010,010101
      112777     124     052
18709                                                                                 ;BINARY DIVIDE FOR 6 BITS
18710                                           .EVEN
18711
```

```
                                        .SBTTL  TEST  - BCSR READ/WRITE BITS
                                        ;THE SECOND TEST WILL CHECK THAT BCSR<7-5;2-0> CAN BE WRITTEN AND
                                        ;READ AS ZEROES AND ONES.  BCSR<14,03> SHOULD BE 0'S.  BCSR<04>
                                        ;SHOULD BE CLEARED BY RESET INSTRUCTION.
                                        ;ROUTINE TEST
                                        ;.      FOR PATTERN=011,010,101 DO
                                        ;.      .       WRITE BCSR<7-5>=PATTERN
                                        ;.      .       IF BCSR<7-5> NE PATTERN THEN
                                        ;.      .       .       ERROR
                                        ;.      .       ENDIF
                                        ;.      .       WRITE BCSR<2-0>=PATTERN
                                        ;.      .       IF BCSR<2-0> NE PATTERN THEN
                                        ;.      .       .       ERROR
                                        ;.      .       ENDIF
                                        ;.      ENDDO
                                        ;.      IF BCSR<14,03> NE <0,0> THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;.      LET BCSR<04>=1
                                        ;.      IF BCSR<04> NE #1 THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;.      EXECUTE "RESET"
                                        ;.      IF BCSR<04> NE 0 THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;.      LET BCSR<04>=0 (THIS BIT IS WRITE ENABLE FOR EAROM)
                                        ;ENDROUTINE

                                        ;;*****************************************************************
        113002  000004                  TST31:  SCOPE
        113004  013737  177520  002730          MOV     BCSR,SAVBR                      ;SAVE BCSR
        113012  005037  177520                  CLR     BCSR                            ;CLEAR BCSR
                                        ;
                                        ; WRITE TO BITS <7-5> AND <2-0>
                                        ;
        113016  012703  113242                  MOV     #THRBIT,R3                      ;POINTER FOR PATTERN TABLE
        113022  005037  001124                  CLR     $GDDAT                          ;CLEAR A LOCATION
        113026  012702  000003                  MOV     #3,R2                           ;DO FOR ALL PATTERNS
        113032  111337  177520          1$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <7-5>
        113036  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN
        113042  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
        113046  001401                          BEQ     2$                              ;IF SO, BRANCH
        113050  104052                          ERROR   +52                             ;ERROR IN BCSR READ/WRITE BITS
        113052  111337  177520          2$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <2-0>
        113056  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN FOR ERRORS
        113062  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
        113066  001401                          BEQ     3$                              ;IF SO, BRANCH
        113070  104052                          ERROR   +52                             ;ERROR IN BCSR READ/WRITE BITS
        113072  077221                  3$:     SOB     R2,1$                           ;CONTINUE TILL ALL PATTERNS DONE
                                        ;
                                        ; CHECK UNUSED BITS <3>
                                        ;
        113074  012737  000010  177520          MOV     #10,BCSR                        ;WRITE TO BIT <3>
        113102  032737  000010  177520          BIT     #BIT03,BCSR                     ;ALL ZEROES?
        113110  001403                          BEQ     4$                              ;IF YES, BRANCH
        113112  005037  001124                  CLR     $GDDAT                          ;EXPECTED PATTERN
```

```
113116  104052                          ERROR   +52                         ;ERROR IN BCSR READ/WRITE BITS
                                ;
                                ; CHECK THAT BIT <4> CLEARS BY RESET
                                ;
113120  052737  000020  177520  4$:     BIS     #BIT04,BCSR                 ;SET BIT 4
113126  032737  000020  177520          BIT     #BIT04,BCSR                 ;WRITTEN OK?
113134  001005                          BNE     5$                          ;IF SO BRANCH
113136  012737  000020  001124          MOV     #BIT04,$GDDAT               ;EXPECTED PATTERN
113144  104052                          ERROR   +52                         ;ERROR IN BCSR READ/WRITE BITS
113146  000415                          BR      7$                          ;EXIT TEST
113150  042737  000020  177520  5$:     BIC     #BIT04,BCSR                 ;TRY TO CLEAR BIT 4
113156  032737  000020  177520          BIT     #BIT04,BCSR                 ;CLEARED OK?
113164  001403                          BEQ     6$                          ;IF SO BRANCH
113166  005037  001124                  CLR     $GDDAT                      ;EXPECTED PATTERN
113172  104052                          ERROR   +52                         ;ERROR IN BCSR READ/WRITE BITS
113174  005737  001206          6$:     TST     $PASS                       ;FIRST PASS?
113200  001014                          BNE     8$                          ;IF  NOT FIRST PASS,EXIT
113202  052737  000020  177520  7$:     BIS     #BIT04,BCSR                 ;SET BIT 4 AGAIN
113210  000005                          RESET                               ;EXECUTE RESET
113212  032737  000020  177520          BIT     #BIT04,BCSR                 ;BIT 4 CLEARED?
113220  001404                          BEQ     8$                          ;IF YES, BRANCH
113222  104053                          ERROR   +53                         ;RESET DOESN'T CLEAR BCSR<4>
113224  042737  000020  177520          BIC     #BIT04,BCSR                 ;CLEAR BIT 4
113232  013737  002730  177520  8$:     MOV     SAVBR,BCSR                  ;RESTORE BCSR
113240  000403                          BR      TST32           ;;GO TO NEXT TEST

113242     140     003     100  THRBIT: .BYTE   140,3,100,2,240,5           ;011,010,101 FOR BITS <7-5,2-0>
113245     002     240     005
```

```
18798                                   .SBTTL  TEST  - 16 BIT ROM CHECKSUM TEST
18799                                   ;ROM'S CHECKSUMS
18800                                   ;
18801                                   ;16 BIT ROM TEST
18802                                   ;THE FIRST TEST WILL CLEAR BCSR<07>, LOAD PCR<14-09> WITH
18803                                   ;ROM ADDRESS BITS <14-09>, AND CHECK CHECKSUMS OF 16-BIT ROM
18804                                   ;BY ACCESSING IT THRU BUS ADDRESSES 173000-173776.  THEN WITH
18805                                   ;BCSR<06;05> BOTH CLEAR, PCR<06-01> USED AS ADDRESS BITS 14-09,
18806                                   ;THE SAME THING WILL BE DONE BY ADDRESSING 16-BIT ROM THRU BUS ADDRESSES
18807                                   ;165000-165776. THE RESULTS SHOULD BE THE SAME AND SHOULD COMPARE
18808                                   ;WITH THAT STORED IN THE BOOT AND DIAGNOSTIC ROM.
18809                                   ;
18810                                   ;BCSR   <07>    DISABLE 17773000
18811                                   ;       <06>    DISABLE 17765000
18812                                   ;       <05>    ROM SOCKET 3 AT 17765000
18813                                   ;
18814                                   ;ROUTINE TEST
18815                                   ;.      LET BCSR<7,6,5>=0,0,0
18816                                   ;.      DO FOR R1 FROM #0 TO #31. BY #1 DO
18817                                   ;.      .       LET PCR<14-09>=R1
18818                                   ;.      .       DO FOR R2 FROM #0 TO #776 BY 2
18819                                   ;.      .       .       CALCULATE CHECKSUM THRU 173000
18820                                   ;.      .       ENDDO
18821                                   ;.      ENDDO
18822                                   ;.      IF CHECKSUM NE #0 THEN
18823                                   ;.      .       ERROR
18824                                   ;.      ENDIF
18825                                   ;.      DO FOR R1 FROM #0 TO #31. BY #1
18826                                   ;.      .       LET PCR<06-01>=R1
18827                                   ;.      .       DO FOR R2 FROM #0 TO #776 BY 2
18828                                   ;.      .       .       CALCULATE CHECKSUM THRU 165000
18829                                   ;.      .       ENDDO
18830                                   ;.      ENDDO
18831                                   ;.      IF CHECKSUM NE #0 THEN
18832                                   ;.      .       ERROR
18833                                   ;.      ENDIF
18834                                   ;ENDROUTINE
18835
18836                                   ;;*****************************************************************************
    113250  000004                      TST32:  SCOPE
18837 113252  013737  177520  002730            MOV     BCSR,SAVBR                      ;SAVE BCSR
18838 113260  042737  000340  177520            BIC     #BIT07!BIT06!BIT05,BCSR         ;READ 16 BIT ROM
18839 113266  052737  001000  177520            BIS     #1000,BCSR                      ; ENABLE HOB, FOR APT
18840                                   ;
18841                                   ; CALCULATE LOW BYTE CHECKSUM'S THRU 173000 AND 165000
18842                                   ;
18843 113274  005001                            CLR     R1                              ;PAGE COUNT FOR ALL 8K
18844 113276  005037  177522                    CLR     PCR                             ;CLEAR PAGE CONTROL REGISTER
18845 113302  005037  002724            1$:     CLR     ACTCHS                          ;CLEAR CHECKSUM AT 173000
18846 113306  005037  001160                    CLR     $TMP0                           ;AT 165000
18847 113312  005002                            CLR     R2                              ;CLEAR COUNTER THRU A PAGE
18848 113314  016203  173000            2$:     MOV     173000(R2),R3                   ;GET LOW BYTE THRU 173000
18849 113320  016204  165000                    MOV     165000(R2),R4                   ;THRU 165000
18850 113324  060337  002724                    ADD     R3,ACTCHS                       ;CALCULATE CHECKSUM THRU 173000
18851 113330  060437  001160                    ADD     R4,$TMP0                        ;CALCULATE CHEKCSUM THRU 165000
18852 113334  005722                            TST     (R2)+                           ;GET NEXT WORD
18853 113336  022702  000776                    CMP     #776,R2                         ;WORD BEFORE LAST?
```

```
18854 113342  001004                        BNE     3$                      ;IF NOT, BRANCH
18855 113344  020337  177522                CMP     R3,PCR                  ;PAGE NUMBER STORED OK?
18856 113350  001401                        BEQ     3$                      ;IF YES, BRANCH
18857 113352  104054                        ERROR   ^54                     ;PAGE NUMBER STORED WRONG
18858 113354  022702  001000          3$:   CMP     #1000,R2                ;LAST WORD IN A PAGE?
18859 113360  003355                        BGT     2$                      ;IF NOT, BRANCH
18860 113362  005737  002724                TST     ACTCHS                  ;CHECKSUM 0?
18861 113366  001401                        BEQ     4$                      ;IF YES, BRANCH
18862 113370  104054                        ERROR   ^54                     ;IN CHECKSUM AT 173000
18863 113372  005737  001160          4$:   TST     $TMP0                   ;CHECKSUM 0 AT 165000?
18864 113376  001401                        BEQ     5$                      ;IF YES, BRANCH
18865 113400  104054                        ERROR   ^54                     ;IN CHECKSUM AT 165000
18866 113402  062701  000002          5$:   ADD     #2,R1                   ;GET NEW PAGE
18867 113406  110137  177522                MOVB    R1,PCR                  ;STORE IN PCR<6-1>
18868 113412  110137  177523                MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
18869 113416  022701  000160                CMP     #160,R1                 ;ALL  PAGES IN R2 FROM BIT1?
18870 113422  003327                        BGT     1$                      ;IF NOT BRANCH
18871                                       ;
18872                                       ; CHECK THE LAST 2K OF ASCII TEXT
18873                                       ;
18874 113424  005037  002724          LASTCH: CLR     ACTCHS                  ;CLEAR CHECKSUM AT 173000
18875 113430  005037  001160                CLR     $TMP0                   ;AT 165000
18876 113434  005002                  1$:   CLR     R2                      ;CLEAR COUNTER THRU A PAGE
18877 113436  016203  173000          2$:   MOV     173000(R2),R3           ;GET LOW BYTE THRU 173000
18878 113442  016204  165000                MOV     165000(R2),R4           ;THRU 165000
18879 113446  060337  002724                ADD     R3,ACTCHS               ;CALCULATE CHECKSUM THRU 173000
18880 113452  060437  001160                ADD     R4,$TMP0                ;CALCULATE CHEKCSUM THRU 165000
18881 113456  005722                        TST     (R2)+                   ;GET NEXT WORD
18882 113460  022702  001000          3$:   CMP     #1000,R2                ;LAST WORD IN A PAGE?
18883 113464  003364                        BGT     2$                      ;IF NOT, BRANCH
18884 113466  022701  000176                CMP     #176,R1                 ;LAST PAGE?
18885 113472  001005                        BNE     4$                      ;IF NOT, BRANCH
18886 113474  023737  173774  177522        CMP     173774,PCR              ;PROPER PAGE NUMBER?
18887 113502  001401                        BEQ     4$                      ;IF YES, BRANCH
18888 113504  104054                        ERROR   ^54                     ;ERROR IN ROM
18889 113506  062701  000002          4$:   ADD     #2,R1                   ;GET NEW PAGE
18890 113512  110137  177522                MOVB    R1,PCR                  ;STORE IN PCR<6-1>
18891 113516  110137  177523                MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
18892 113522  022701  000200                CMP     #200,R1                 ;ALL  PAGES IN R2 FROM BIT1?
18893 113526  003342                        BGT     1$                      ;IF NOT BRANCH
18894 113530  005737  002724                TST     ACTCHS                  ;CHECKSUM 0?
18895 113534  001401                        BEQ     5$                      ;IF YES, BRANCH
18896 113536  104054                        ERROR   ^54                     ;IN CHECKSUM AT 173000
18897 113540  005737  001160          5$:   TST     $TMP0                   ;CHECKSUM 0 AT 165000?
18898 113544  001401                        BEQ     6$                      ;IF YES, BRANCH
18899 113546  104054                        ERROR   ^54                     ;IN CHECKSUM AT 165000
18900 113550  005037  177522          6$:   CLR     PCR                     ;CLEAR PCR
18901 113554  042737  001000  177520        BIC     #1000, BCSR             ; RESET MOB FOR APT
18902 113562  013737  177520  002730        MOV     BCSR,SAVBR              ;RESTORE BCSR
18903
18904
18905
```

TEST - 8 BIT EEROM CHECKSUM (105DEC BYTES) TEST

```
18907                                   .SBTTL  TEST  - 8 BIT EEROM CHECKSUM (105DEC BYTES) TEST
18908                                   ;THIS TEST WILL CLEAR BCSR<6>, SET BCSR<5>, LOAD PCR<5-1>
18909                                   ;WITH ADDRESS BITS 13-09, AND CHECK CRC PATTERNS OF 8-BIT EEROM BY
18910                                   ;ACCESSING IT THRU ADDRESSES 165000-165776. THIS TEST VERIFIES THE
18911                                   ;RESPONSE ONLY OF THE BASE AREA.
18912                                   ;ROUTINE TEST
18913                                   ;.      LET BCSR<5>=1
18914                                   ;.      LET OLDCRC=#0
18915                                   ;.      LET PCR<05-01>=#0
18916                                   ;.      CALCULATE CHECKSUM FOR THE FIRST 320 LOCATIONS
18917                                   ;.      IF RESULTING CHECKSUM NOT ZERO THEN
18918                                   ;.          ERROR
18919                                   ;.      ENDIF
18920                                   ;ENDROUITNE
18921
18922                                   ;;**********************************************************************
     113570  000004                    TST33:  SCOPE
18923 113572  013737  177520  002730          MOV     @CSR,SAVBR                      ;SAVE BCSR
18924 113600  042737  000100  177520          BIC     #BIT06,@CSR                     ;ENABLE INTERNAL RESPONSE
18925 113606  052737  000040  177520          BIS     #BIT05,@CSR                     ;SELECT 8-BIT ROM
18926 113614  005037  001160                  CLR     $TMP0                           ;CLEAR SUM
18927                                   ;
18928                                   ; CALCULATE LOW BYTE CHECKSUM'S THRU 165000
18929                                   ;
18930 113620  005001                    1$:     CLR     R1                              ;PAGE COUNT FOR ALL 8K
18931 113622  005037  177522                    CLR     PCR                             ;CLEAR PAGE CONTROL REGISTER
18932 113626  005037  002724            2$:     CLR     ACTCHS                          ;CLEAR CHECKSUM AT 165000
18933 113632  005002                            CLR     R2                              ;CLEAR COUNTER THRU A PAGE
18934 113634  116204  165000            3$:     MOVB    165000(R2),R4                   ;GET A BYTE THRU 165000
18935 113640  060437  001160                    ADD     R4,$TMP0                        ;CALCULATE CHEKCSUM THRU 165000
18936 113644  005722                            TST     (R2)+                           ;GET NEXT WORD
18937 113646  022702  000316                    CMP     #316,R2                         ;WORD BEFORE LAST?
18938 113652  001004                            BNE     4$                              ;IF NOT, BRANCH
18939 113654  122704  000252                    CMPB    #252,R4                         ;314 SHOULD HAVE 252
18940 113660  001765                            BEQ     3$                              ;IF YES, BRANCH
18941 113662  104055                            ERROR   .55                             ;PAGE NUMBER STORED WRONG
18942 113664  022702  000322            4$:     CMP     #322,R2                         ;LAST WORD IN A PAGE?
18943 113670  003361                            BGT     3$                              ;IF NOT, BRANCH
18944 113672  113737  165010  001162          MOVB    165010,$TMP1                    ;STORE SIZE,<3>=1 2K
18945 113700  105737  001160                  TSTB    $TMP0                           ;CHECKSUM 0 AT 165000?
18946 113704  001401                          BEQ     5$                              ;IF YES, BRANCH
18947 113706  104055                          ERROR   .55                             ;IN CHECKSUM AT 165000
18948 113710  005037  177522            5$:     CLR     PCR
18949 113714  013737  002730  177520          MOV     SAVBR,@CSR                      ;RESTORE BCSR
18950
```

```
                                        .SBTTL  TEST - 8-BIT EAROM READ-WRITE - TEST

                                        ; THIS TEST WRITES A CHECKERBOARD PATTERN OF ALTERNATING 1'S AND 0'S IN
                                        ; 8-BIT EAROM.
                                        ; ALL BASE AREA 316 OCTAL LOCATIONS ARE SAVED. (105 DEC PLUS 4 CUSTOMER LOC.)
                                        ; IT WILL RUN ONLY IF BIT 12 IS SET IN SOFTWARE SWITCH REGISTER.
                                        ; THIS IS THE EEROM TEST ENABLE, SETABLE AT RUNTIME BY AN OPERATOR.  LOOP
                                        ; ON ERROR IS PROHIBITED, AS IT WOULD BE POSSIBLE TO DESTROY THE EEROM
                                        ; IN A RELATIVLY SHORT TIME WITH REPEATED WRITE CYCLES.  THE OPTION TO
                                        ; CONTINUE BEYOND A SINGLE ERROR IS SELECTED BY BY ASSERTING THE HALT ON
                                        ; ERROR SWITCH AT WHICH TIME THE OPERATOR MAY TRUNCATE THE TEST.
                                        ;
                                        ; THE SWITCHES WHICH COULD AFFECT THE EEROM'S LIFE ARE MASKED OUT AND
                                        ; RESTORED IN THE TEST IMMEDIATLY FOLLOWING THIS ONE. <<<<<********
                                        ;
                                        ; THIS TEST WILL BE NOT BE DONE IF THERE IS AN APT SYSTEM PRESENT, OR IF
                                        ; BIT #12 OF THE SOFTWARE SWITCH REGISTER IS NOT SET.  IF BIT 12 IS SET AND
                                        ; THERE IS NO APT SYSTEM PRESENT (OR IN LOAD MODE), THE EEROM TEST WILL BE
                                        ; RUN 1 TIME, ALL SUBSEQUENT PASSES WILL NOT INCLUDE THE EEROM TEST.
                                        ;
                                        ;
                                        ;                        -- SECTION 1
                                        ;+  SAVE BASE AREA
                                        ;+
                                        ;+
                                        ;+
                                        ;+ TEST                  -- SECTION 2
                                        ;+
                                        ;+ 2K PROCEDURE
                                        ;+   DO WHILE NOT$DONE$2000
                                        ;+      WRITE 10101010 (252 OCT)
                                        ;+      DELAY
                                        ;+      READ  10101010 (252 OCT)
                                        ;+      UPDATE ERROR LIST IF INCORRECT READ
                                        ;+   ENDDO
                                        ;+   DO WHILE NOT$DONE$2000
                                        ;+      WRITE 01010101 (125 OCT)
                                        ;+      DELAY
                                        ;+      READ  01010101 (125 OCT)
                                        ;+      UPDATE ERROR LIST
                                        ;+   ENDDO
                                        ;+

                                        ; CLEANUP -- SECTION 3
                                        ; COMMUNICATE FAILURE MODE AND LOC INFO
                                        ; RESTORE BASE AREA (105) + 4 CUSTOMER LOCATIONS
                                        ; RESTORE REGISTERS
                                        ; RESTORE BACK BASE AREA
                                        ;
                                        ;
                                        ;*******************************************************************************

                                        ;;*******************************************************************
     113722  000004                     TST34:  SCOPE
     113724  017737  065210  003050             MOV     @SWR,SAVSWR             ;SAVE SOFTSWITCH SETTINGS
     113732  005737  001206                     TST     $PASS                   ;DO ONLY ONCE, IFF ALL COND. O.K.
     113736  001006                             BNE     1$                      ;  ON FIRST PASS ( 0)
```

```
113740  000240                          NOP                             ; NO, DO AGAINA
                                ;       CMPB    #APTENV,$ENV            ;IN APT MODE?
                                ;       BEQ     1$                      ; <IF YES, EXIT TEST>
113742  032777  010000  065170          BIT     #BIT12,@SWR             ; BIT 12 (DO EEROM TEST) SET?
113750  001016                          BNE     2$                      ;
113752  000240                          NOP
113754  000137  114304          1$:     JMP     TSTEND                  ; EXIT POINT FOR PROGRAM
113760     015     012     116  33$:    .ASCIZ  <15><12>/NOW TESTING EEROM/<15><12>
113763     117     127     040
113766     124     105     123
113771     124     111     116
113774     107     040     105
113777     105     122     117
114002     115     015     012
114005     000
                                        .EVEN
                                ; SAVE BASE AREA, SWITCH REGISTERS

114006  104401  113760          2$:     TYPE    ,33$                    ; INDICATE EEROM TEST UNDERWAY
114012  017737  065122  003050          MOV     @SWR,SAVSWR             ; SAVE SOFTSWITCH SETTINGS
114020  042777  041777  065112          BIC     #41777, @SWR            ; CLEAR ANY INTERFERRING
114026  013737  177520  002730          MOV     BCSR,SAVBR              ;SAVE REGISTER
114034  052737  001060  177520          BIS     #1060,BCSR              ;ENABLE INTERNAL ROM'S
114042  000240                          NOP                             ;ENABLE 8-BIT ROM, HOB
114044  012701  141070                  MOV     #$POWER+10,R1           ;LAST LOCATION IN PROGRAM
114050  005037  177522                  CLR     PCR                     ;START WITH PAGE 0
114054  005002                          CLR     R2                      ;DISPLACEMENT 0
114056  016221  165000          3$:     MOV     165000(R2),(R1)+        ;STORE A WORD
114062  005722                          TST     (R2)+                   ;GET NEXT WORD
114064  022702  000334                  CMP     #334,R2                 ;ALL 332 LOCATIONS DONE?
114070  003372                          BGT     3$                      ;IF NOT, CONTINUE

                                ; TEST 2K SECTION

114072  112703  000252                  MOVB    #252, R3                ; FIRST PATTERN, 10 101 010
114076  005037  003022                  CLR     ERRCNT                  ; ZERO THE CUMULATIVE ERROR COUNT

114102  000240                  201$:   NOP
114104  005037  177522                  CLR     PCR                     ; CLEARS PAGE REGISTER (START @ 165000)
114110  000240                  202$:   NOP
114112  005002                          CLR     R2                      ; FIRST LOCATION EACH PAGE OF EEROM
114114  000240                  203$:   NOP
114116  110362  165000                  MOVB    R3, 165000(R2)          ; WRITE THE TEST PATTERN
114122  004737  114270                  JSR     PC, DELAY               ; WAIT FOR WRITE TIME
114126  120362  165000                  CMPB    R3,165000(R2)           ; READ BACK THE WRITTEN WORD
114132  001413                          BEQ     204$                    ; IF O. K. READBACK, SKIP HANDLE ERROR

114134  005237  003022                  INC     ERRCNT                  ; UPDATE CUMULATIVE ERROR COUNT
114140  116237  165000  001126          MOVB    165000(R2),$BDDAT       ; PUT THE READ DATA IN DISPLAY AREA
114146  013704  177522                  MOV     PCR,R4
114152  010237  001122                  MOV     R2,$BDADR               ; ALSO ADDRESS LOCATION INFO
114156  104134                          ERROR   +134                    ; DO EEROM READ/WRITE ERROR REPORT
114160  000240                          NOP                             ; CONTINUE TESTING

114162  005722                  204$:   TST     (R2)+
114164  022702  000776                  CMP     #776,R2                 ; LAST LOCATION CHECKED IN A PAGE
114170  003351                          BGT     203$                    ; CONTINUE TILL ALL PAGE LOC. TESTED
```

TEST - 8-BIT EAROM READ-WRITE - TEST

```
19058 114172  062737  000002  177522          ADD     #2,PCR                  ; CHANGE PCR EVERY 512 DEC. BYTES
19059 114200  122737  000020  177522          CMPB    #20,PCR                 ; 8 (256 BYTE PGS IN 2K) * 2 (HOLES)=20
19060 114206  001340                          BNE     202$                    ; FINISH PAGE
19061 114210  122703  000125                  CMPB    #125,R3                 ; TEST FOR BOTH PATTERNS WRITTEN
19062 114214  001403                          BEQ     205$                    ; EXIT POINT
19063 114216  112703  000125                  MOVB    #125,R3                 ; INVERT THE TEST PATTERN
19064 114222  000727                          BR      201$                    ; DO THE TEST OVER WITH NEW PATTERN
19065
19066
19067 114224  000240                  205$:   NOP                             ; TEST OVER
19068
19069 114226  012701  141070                  MOV     #$POWER+10,R1           ;LAST LOCATION IN PROGRAM
19070 114232  005037  177522                  CLR     PCR                     ;START WITH PAGE 0
19071 114236  005002                          CLR     R2                      ;DISPLACEMENT 0
19072 114240  012162  165000          11$:    MOV     (R1)+,165000(R2)        ;RESTORE A WORD
19073 114244  004737  114270                  JSR     PC, DELAY               ; WAIT FOR WRITE TIME
19074 114250  005722                          TST     (R2)+                   ;GET NEXT WORD
19075 114252  022702  000334                  CMP     #334,R2                 ;ALL 332 LOCATIONS DONE?
19076 114256  003370                          BGT     11$                     ;IF NOT, CONTINUE
19077 114260  013737  002730  177520          MOV     SAVBR,BCSR              ;RESTORE REGISTER
19078 114266  000406                          BR      TSTEND                  ; GOTO NEXT TEST (SWR RESTORED THERE)
19079
19080                         ; -------------- SUBROUTINES AREA ----------------
19081                         ; THE FOLLOWING SUBROUTINE CAUSES A DELAY OF A CERTAIN NO OF MILLISECONDS
19082                         ; THE NUMBER OF MS. DELAYED IS BASED ON THE TYPE OF EEROM AS INDICATED BY R5
19083
19084 114270  010046          DELAY:  MOV     R0,-(SP)                ; SAVE R0
19085 114272  012700  023420          MOV     #10000., R0             ; ALL ELSE = 10 MS DELAY
19086 114276  077001          1$:     SOB     R0, 1$                  ; ACTUAL DELAY
19087 114300  012600                  MOV     (SP)+, R0               ; RESTORE R0
19088 114302  000207                  RTS     PC
19089
19090                         ;-------------------------------------------------------
19091 114304  000240          TSTEND: NOP                             ; EVERYTHING DONE
19092
```

TEST - LKS BIT 7

```
19094                                   .SBTTL  TEST  - LKS BIT 7
19095                                   ; THIS TEST RESTORES THE SOFTWARE SWITCH REGISTER USED IN THE EEROM
19096                                   ; TEST ABOVE, AND THEREFORE MUST FOLLOW IT DIRECTLY !! <<<<<<<<<*****
19097                                   ;THESE TESTS WILL CHECK FUNCTIONALITY OF LKS<7-6>
19098                                   ;
19099                                   ;ACCESS LKS, CHECK THAT READY LKS<07> CAN BE 0 AND 1.
19100                                   ;
19101                                   ;LKS     <07>     LINE CLOCK MONITOR
19102                                   ;        <06>     LINE CLOCK INTERRUPT ENABLE
19103                                   ;ROUTINE TEST
19104                                   ;IF UFD AND LKS IS DISABLED THEN
19105                                   ;.       EXIT TEST
19106                                   ;ENDIF
19107                                   ;.       READ LKS TO SEE IF IT TIMEOUTS
19108                                   ;.(CHECK LKS<07>)
19109                                   ;.       LET R1=#77777(WORST CASE COUNTER FOR SLOW CLOCK)
19110                                   ;.       LET 100=#ADDRESS OF LINE_CLOCK_INTERRUPT
19111                                   ;.       CLEAR INTERRUPT_FLAG
19112                                   ;.       DO 3 TIMES
19113                                   ;.       .        REPEAT
19114                                   ;.       .        .        DECREMENT R1
19115                                   ;.       .        UNTIL R1 NE #0 OR LOW BYTE OF LKS LT #0
19116                                   ;.       .        IF LOW BYTE OF LKS GE #0 THEN (READY DIDN'T COME UP)
19117                                   ;.       .        .        ERROR
19118                                   ;.       .        ENDIF
19119                                   ;.       ENDDO
19120                                   ;.       IF INTERRUPT_FLAG NE #0 THEN (INTERUPT W/O LKS<6>=1)
19121                                   ;.       .        ERROR
19122                                   ;.       ENDIF
19123                                   ;ENDROUTINE
19124
19125                                   ;;*****************************************************************
    114306  000004                      TST35:  SCOPE
19126 114310  013777  003050  064622            MOV     SAVSWR,@SWR                     ; RESTORE HERE FROM EEROM TEST
19127 114316  032737  000100  000052            BIT     #BIT06,@#52                     ;UFD MODE?
19128 114324  001404                            BEQ     1$                              ;IF NOT, GO DO TEST
19129 114326  032737  010000  177520            BIT     #BIT12,BCSR                     ;LKS DISABLED?
19130 114334  001056                            BNE     TST36           ;;IF DISABLED, EXIT TEST
19131                                   ;
19132                                   ; TRY TO ACCESS LKS WITHOUT TIMEOUT
19133                                   ;
19134 114336  013737  000004  001160    1$:     MOV     ERRVEC,$TMP0                    ;SAVE TIMEOUT VECTOR
19135 114344  012737  114366  000004            MOV     #2$,ERRVEC                      ;POINT NEW VECTOR TO PROGRAM
19136 114352  012737  000340  000006            MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
19137 114360  005737  177546                    TST     LKS                             ;ACCESS LKS
19138 114364  000403                            BR      3$                              ;IF NO TIMEOUT, CONTINUE
19139 114366  104056                    2$:     ERROR   .56                             ;TIMEOUT READING LKS
19140 114370  005726                            TST     (SP)+                           ;ADJUST STACK POINTER
19141 114372  005726                            TST     (SP)+
19142 114374  013737  001160  000004    3$:     MOV     $TMP0,ERRVEC                    ;RESTORE TIMEOUT VECTOR
19143                                   ;
19144                                   ; CHECK THAT READY BIT LKS<7> CAN BE 1
19145                                   ;
19146 114402  042737  000200  177546            BIC     #BIT07,LKS                      ; START WITH CLEAN SLATE
19147 114410  005037  002722                    CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
19148 114414  012737  134376  000100            MOV     #LKSINT,100                     ;POINT VECTOR TO INERRUPT ROUTINE
19149 114422  012737  000340  000102            MOV     #340,102                        ;AT PRIORITY 7
```

TEST - LKS BIT 7

```
19150 114430  012702  000003            MOV     #3,R2                           ;DO 3 TIMES TO SYNCHRONISE
19151 114434  012701  077777      4$:   MOV     #77777,R1                       ;COUNTER FOR SLOW CLOCKS
19152 114440  105737  177546      5$:   TSTB    LKS                             ;READY LKS<7>=1?
19153 114444  100401                    BMI     6$                              ;IF SO, DO NEXT LOOP
19154 114446  077104                    SOB     R1,5$                           ;OTHERWISE, GO THRU COUNT
19155 114450  105737  177546      6$:   TSTB    LKS                             ;WAS READY 1?
19156 114454  100401                    BMI     7$                              ;IF YES, BRANCH
19157 114456  104057                    ERROR   .57                             ;LKS<07> DOES NOT BECOME 1
19158 114460  077213                7$: SOB     R2,4$                           ;DO ALL 3 TIMES
19159 114462  005737  002722      8$:   TST     LKSFL                           ;ANY INTERRUPTS W/O LKS<6>=1?
19160 114466  001401                    BEQ     TST36           ;;IF NONE, EXIT TEST
19161 114470  104061                    ERROR   .61                             ;ILLEGAL CLOCK INTERRUPTS
19162
```

```
19164                                   .SBTTL  TEST  - LKS INTERRUPT PRIORITY
19165                                   ;CHECK THAT LKS INTERRUPTS HAPPEN AT PRIORITY 5 CLEARING LKS<07>
19166                                   ;AND DON'T HAPPEN AT PRIORITY 6.
19167                                   ;ROUTINE TEST
19168                                   ;IF UFD AND LKS IS DISABLED THEN
19169                                   ;.      EXIT TEST
19170                                   ;ENDIF
19171                                   ;.      SET PRIORITY TO 5
19172                                   ;.      CLEAR INTERRUPT_FLAG
19173                                   ;.      LET LKS<06>=#1 (ENABLE INTERRUPTS)
19174                                   ;.      SET COUNTER TO WAIT FOR 3 INTERRUPTS
19175                                   ;.      REPEAT
19176                                   ;.      .       DECREMENT COUNTER
19177                                   ;.      UNTIL INTERRUPT_FLAG EQ #3 OR COUNTER EQ #0
19178                                   ;.      CLEAR LKS<06>
19179                                   ;.      IF LKS<07> EQ #1 THEN
19180                                   ;.      .       ERROR (WAS NOT CLEARED ON INTERRUPT)
19181                                   ;.      ENDIF
19182                                   ;.      IF COUNTER LT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR_800HZ
19183                                   ;.      .       ERROR (INTERRUPTS NEVER GO LOW)
19184                                   ;.      ENDIF
19185                                   ;.      IF INTERRUPT_FLAG LT #3 THEN
19186                                   ;.      .       ERROR (INTERRUPTS DON'T HAPPEN)
19187                                   ;.      ENDIF
19188                                   ;.      CLEAR INTERUPT_FLAG
19189                                   ;.      WAIT FOR LKS<7>=1
19190                                   ;.      LET LKS<7>=0
19191                                   ;.      IF LKS<7> NE #0 THEN
19192                                   ;.      .       ERROR (LKS<7> NOT CLEARED
19193                                   ;.      ENDIF
19194                                   ;.      SET PRIORITY TO 6
19195                                   ;.      SET COUNTER TO 1 SLOW CLOCK INTERRUPT
19196                                   ;.      SET LKS<06>
19197                                   ;.      REPEAT
19198                                   ;.      .       DECREMENT COUNTER
19199                                   ;.      UNTIL COUNTER EQ #0 OR INTERRUPT_FLAG NE #0
19200                                   ;.      IF INTERRUPT_FLAG NE #0 THEN
19201                                   ;.      .       ERROR (INTERRUPT WAS AT WRONG PRIORITY)
19202                                   ;.      ENDIF
19203                                   ;.      RESTORE ORIGINAL PRIORITY
19204                                   ;ENDROUTINE
19205                                   ;
19206                                   ;ROUTINE LINE_CLOCK_INTERUPT
19207                                   ;.      INCREMENT INTERRUPT_FLAG
19208                                   ;ENDROUTINE
19209
19210                                   ;;*****************************************************************
     114472  000004                   TST36:  SCOPE
19211 114474  032737  000100  000052           BIT     #BIT06,@#52                     ;UFD MODE?
19212 114502  001404                           BEQ     1$                              ;IF NOT, GO DO TEST
19213 114504  032737  010000  177520           BIT     #BIT12,BCSR                     ;LKS DISABLED?
19214 114512  001132                           BNE     TST37           ;;IF DISABLED, EXIT TEST
19215                                   ;
19216                                   ; WAIT FOR 3 INTERRUPTS AND CHECK LKS<7> TO BE 0 AFTER INTERRUPT
19217                                   ;
19218 114514  042737  000100  177546   1$:     BIC     #BIT06,LKS                      ; FROM END OF TEST 42?? PROBLEM
19219 114522  005037  002722                   CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
```

TEST - LKS INTERRUPT PRIORITY

```
19220 114526  012737  134376  000100          MOV     #LKSINT,100             ;POINT VECTOR TO ROUTINE
19221 114534  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
19222 114540  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19223 114546  032737  000100  177546          BIT     #BIT06,LKS              ;BIT SET OK?
19224 114554  001001                          BNE     2$                      ;IF YES, BRANCH
19225 114556  104131                          ERROR   +131                    ;ERROR WRITING 1 TO LKS<6>
19226 114560  106427  000240          2$:     MTPS    #240                    ;SET PRIORITY TO 5
19227 114564  022737  000003  002722  3$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
19228 114572  001401                          BEQ     4$                      ;IF YES, BRANCH
19229 114574  077105                          SOB     R1,3$                   ;STAY IN A LOOP
19230                                 ;
19231                                 ; DISABLE INTERRUPTS AND CHECK THAT PROPER CONDITIONS ARE MET
19232                                 ;
19233 114576  042737  001000  177520  4$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
19234 114604  106427  000340                  MTPS    #340                    ;RAISE PRIORITY
19235 114610  042737  000100  177546          BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
19236 114616  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED AFTER INTERRUPTS?
19237 114624  001401                          BEQ     5$                      ;IF 0, BRANCH
19238 114626  104062                          ERROR   +62                     ;INTERRUPTS DON'T CLEAR LKS<7>
19239 114630  105737  177546          5$:     TSTB    LKS                     ;LKS<7>=1?
19240 114634  100375                          BPL     5$                      ;IF NOT, WAIT
19241 114636  005037  177546                  CLR     LKS                     ;CLEAR LKS<7>
19242 114642  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED?
19243 114650  001401                          BEQ     6$                      ;IF YES, BRANCH
19244 114652  104060                          ERROR   +60                     ;LKS<7> NOT CLEARED ON WRITE
19245 114654  032737  000100  177546  6$:     BIT     #BIT06,LKS              ;LKS<6>=0?
19246 114662  001401                          BEQ     7$                      ;IF YES, BRANCH
19247 114664  104131                          ERROR   +131                    ;ERROR WRITING 0 TO LKS<6>
19248 114666  052737  001000  177520  7$:     BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
19249 114674  022701  077737                  CMP     #77737,R1               ;COUNTER AT LESS THAN 800HZ?
19250 114700  002001                          BGE     8$                      ;IF NOT, BRANCH
19251 114702  104063                          ERROR   +63                     ;READY LINE DOES NOT GO LOW
19252 114704  022737  000003  002722  8$:     CMP     #3,LKSFL                ;DID 3 INTERRUPTS HAPPEN?
19253 114712  001404                          BEQ     9$                      ;IF YES, BRANCH
19254 114714  012737  000003  001124          MOV     #3,$GDDAT               ;3 INTERRUPTS EXPECTED
19255 114722  104064                          ERROR   +64                     ;INTERRUPTS DON'T HAPPEN
19256                                 ;
19257                                 ; CHECK WHETHER INTERRUPTS HAPPEN AT PRIORITY 6
19258                                 ;
19259 114724  005037  002722          9$:     CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
19260 114730  106427  000300                  MTPS    #300                    ;RAISE PRIORITY TO 6
19261 114734  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
19262 114740  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19263 114746  005737  002722          10$:    TST     LKSFL                   ;ANY INTERRUPTS?
19264 114752  001001                          BNE     11$                     ;IF YES, EXIT LOOP
19265 114754  077104                          SOB     R1,10$                  ;CONTINUE WITH COUNT
19266 114756  005737  002722          11$:    TST     LKSFL                   ;ANY INTERRUPTS?
19267 114762  001404                          BEQ     12$                     ;IF NO, BRANCH
19268 114764  012737  000005  001124          MOV     #5,$GDDAT               ;STORE PRIORITY FOR TYPE OUT
19269 114772  104065                          ERROR   +65                     ;INTERUPTS HAPPEN AT WRONG PRIORITY
19270 114774  106427  000340          12$:    MTPS    #340                    ;RESTORE PRIORITY
19271
```

```
                                        .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
                                        ;

                                        ;;*****************************************************************
        115000  000004                  TST37:  SCOPE
        115002  032737  000100  000052          BIT     #BIT06,@#52                     ;UFD MODE?
        115010  001404                          BEQ     1$                              ;IF NOT, BRANCH
        115012  032737  010000  177520          BIT     #BIT12,BCSR                     ;LKS DISABLED?
        115020  001052                          BNE     TST40           ;;IF DISABLED, EXIT TEST
                                        ;
                                        ; CHECK BCSR<12> TO BE 0 AND 1
                                        ;
        115022  013737  177520  002730  1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
        115030  042737  010000  177520          BIC     #BIT12,BCSR                             ;CLEAR BCSR
        115036  032737  010000  177520          BIT     #BIT12,BCSR                     ;<12>=0?
        115044  001403                          BEQ     2$                              ;IF OK, BRANCH
        115046  005037  001124                  CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
        115052  104052                          ERROR   +52                             ;ERROR BCSR READ/WRITE BITS
        115054  052737  010000  177520  2$:     BIS     #BIT12,BCSR                     ;SET BIT 12
        115062  032737  010000  177520          BIT     #BIT12,BCSR                     ;GOT SET OK?
        115070  001004                          BNE     3$                              ;IF OK, BRANCH
        115072  012737  010000  001124          MOV     #BIT12,$GDDAT                   ;EXPECTED PATTERN
        115100  104052                          ERROR   +52                             ;ERROR BCSR READ/WRITE BITS
                                        ;
                                        ; TRY TO ACCESS LKS TO GET A TIMEOUT WITH BCSR<12>=1
                                        ;
```

TEST - LINE CLOCK DISABLE

```
 19329 115102  013701  000004          3$:     MOV     ERRVEC,R1               ;SAVE TIMEOUT VECTOR
 19330 115106  012737  115130  000004          MOV     #4$,ERRVEC              ;POINT TO PROGRAM AREA
 19331 115114  012737  000340  000006          MOV     #340,ERRVEC+2           ;AT PRIORITY 7
 19332 115122  005737  177546                  TST     LKS                     ;ACCESS LKS REGISTER
 19333 115126  104066                          ERROR   +66                     ;BCSR<12> DOES NOT DISABLE LKS
 19334 115130  005726                  4$:     TST     (SP)+                   ;RESTORE STACK POINTER
 19335 115132  005726                          TST     (SP)+
 19336 115134  010137  000004                  MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
 19337 115140  013737  002730  177520          MOV     SAVBR,BCSR              ;RESTORE BCSR
 19338
 19339
```

```
                                .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

                                ;;*****************************************************************
    115146  000004              TST40:  SCOPE
19379 115150  032737  000100  000052        BIT     #BIT06,@#52                     ;UFD MODE?
19380 115156  001404                        BEQ     1$                              ;IF NOT, BRANCH
19381 115160  032737  020000  177520        BIT     #BIT13,BCSR                     ;FORCE INTERRUPT SET?
19382 115166  001530                        BEQ     TST41           ;;IF SET, EXIT TEST
19383                           ;
19384                           ; CHECK BSCR<13> TO BE 0 AND 1
19385                           ;
19386 115170  013737  177520  002730  1$:   MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
19387 115176  042737  001000  177520        BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
19388 115204  042737  020000  177520        BIC     #BIT13,BCSR                     ;CLEAR BCSR
19389 115212  032737  020000  177520        BIT     #BIT13,BCSR                     ;<13>=0?
19390 115220  001403                        BEQ     2$                              ;IF OK, BRANCH
19391 115222  005037  001124                CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
19392 115226  104052                        ERROR   +52                             ;ERROR BCSR READ/WRITE BITS
19393 115230  052737  020000  177520  2$:   BIS     #BIT13,BCSR                     ;SET BIT 13
19394 115236  032737  020000  177520        BIT     #BIT13,BCSR                     ;GOT SET OK?
19395 115244  001004                        BNE     3$                              ;IF OK, BRANCH
19396 115246  012737  020000  001124        MOV     #BIT13,$GDDAT                   ;EXPECTED PATTERN
```

TEST - UNCONDITIONAL CLOCK LINE INTERRUPTS

```
19397 115254  104052                          ERROR   +52                     ;ERROR BCSR READ/WRITE BITS
19398                                 ;
19399                                 ; SET UP  TO DO UNCONDITIONAL INTERRUPTS
19400                                 ;
19401 115256  012737  134376  000100  3$:     MOV     #LKSINT,@#100           ;SET UP INTERRUPT VECTOR
19402 115264  012737  000340  000102          MOV     #340,@#102              ;AT PRIORITY 7
19403 115272  052737  010000  177520          BIS     #BIT12,BCSR             ;FOR 1ST TIME DISABLE LKS
19404 115300  000403                          BR      5$                      ;GO DO IT
19405 115302  042737  010000  177520  4$:     BIC     #BIT12,BCSR             ;FOR THE 2ND TIME, ENABLE LKS
19406 115310  005037  002722          5$:     CLR     LKSFL                   ;CLEAR INTERRUPTS FLAG
19407 115314  012702  077777                  MOV     #77777,R2               ;COUNTER TO WAIT FOR INTERRUPTS
19408 115320  106427  000240                  MTPS    #240                    ;LOWER PRIORITY TO 5
19409 115324  022737  000003  002722  6$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
19410 115332  001401                          BEQ     7$                      ;EXIT LOOP, IF SO
19411 115334  077205                          SOB     R2,6$                   ;OTHERWISE, KEEP WAITING
19412 115336  106427  000340          7$:     MTPS    #340                    ;RAISE PRIORITY TO 7
19413 115342  022702  077700                  CMP     #77700,R2               ;INTERRUPTS HAPPEN TOO OFTEN?
19414 115346  002001                          BGE     8$                      ;IF NOT, BRANCH
19415 115350  104063                          ERROR   +63                     ;READY LINE DOESN'T GO LOW
19416 115352  022737  000003  002722  8$:     CMP     #3,LKSFL                ;AT LEAST 3 INTERRUPTS HAPPENED?
19417 115360  002004                          BGE     9$                      ;IF SO, BRANCH
19418 115362  012737  000003  001124          MOV     #3,$GDDAT               ;EXPECTED DATA
19419 115370  104064                          ERROR   +64                     ;INTERRUPTS DON'T HAPPEN
19420 115372  032737  010000  177520  9$:     BIT     #BIT12,BCSR             ;SECOND TIME THRU THE LOOP?
19421 115400  001340                          BNE     4$                      ;IF NOT, DO IT AGAIN
19422 115402  032737  000100  000052          BIT     #BIT06,@#52             ;UFD MODE?
19423 115410  001404                          BEQ     10$                     ;IF NOT, BRANCH
19424 115412  032737  010000  177520          BIT     #BIT12,BCSR             ;IF UFD AND LKS DISABLED?
19425 115420  001010                          BNE     12$                     ;DON'T CHECK LKS
19426 115422  032737  000100  177546  10$:    BIT     #BIT06,LKS              ;INTERRUPT ENABLE LINE HOLD 1?
19427 115430  001001                          BNE     11$                     ;IF SO, BRANCH
19428 115432  104067                          ERROR   +67                     ;BCSR<13> DOESN'T SET LKS<6>
19429 115434  042737  000100  177546  11$:    BIC     #BIT06,LKS              ;DISABLE LKS INTERRUPTS
19430 115442  013737  002730  177520  12$:    MOV     SAVBR,BCSR              ;RESTORE BCSR
19431
19432
```

```
                                .SBTTL  TEST  - RESETTING LKS
                                ;RESETTING LKS(*)
                                ;THIS TEST WILL PROVE THAT RESET INSTRUCTION SETS LKS<07> AND
                                ;CLEARS LKS<06>.
                                ;ROUTINE TEST
                                ;IF UFD AND LKS IS DISABLED THEN
                                ;.      EXIT TEST
                                ;ENDIF
                                ;.      POINT LKS VECTOR 100 TO ERROR_LKS_ILLEGAL_INTERRUPT
                                ;.      SYNCHRONIZE LKS BY WAITING FOR 3 PULSES
                                ;.      LET LKS<06>=#1
                                ;.      CLEAR LKS (CLEARS LKS<07>
                                ;.      EXECUTE "RESET"
                                ;.      IF LKS<7> NE #1 OR LKS<6> NE #0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      IF ILLEGAL_LINE_CLOCK_INTERRUPT NE 0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;ENDROUTINE
                                ;
                                ;ROUTINE ERROR_LKS_ILLEGAL_INTERRUPT
                                ;.      FLAG ILLEGAL_LINE_CLOCK_INTERRUPT
                                ;RETURN

                                ;;*****************************************************************
115450  000004                  TST41:  SCOPE
115452  005737  001206                  TST     $PASS                           ;FIRST PASS?
115456  001057                  BNE     TST42           ;;IF NOT FIRST PASS, EXIT TEST
115460  032737  000100  000052  BIT     #BIT06,@#52                     ;UFD MODE?
115466  001404                  BEQ     1$                              ;IF NOT, BRANCH
115470  032737  010000  177520  BIT     #BIT12,BCSR                     ;LKS DISABLED?
115476  001447                  BEQ     TST42           ;;IF DISABLED, EXIT TEST
                                ;
                                ; SYNCHRONISE WITH LINE TIME CLOCK BY WAITING FOR 3 INTERRUPTS
                                ;
115500  013737  177520  002730  1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR
115506  042737  010000  177520          BIC     #BIT12,BCSR                     ;ENABLE LKS RESPONSE
115514  012737  134376  000100          MOV     #LKSINT,@#100                   ;SET UP INTERRUPT VECTOR
115522  012737  000340  000102          MOV     #340,@#102                      ;AT PROIRITY 7
115530  052737  000100  177546          BIS     #BIT06,LKS                      ;SET INTERRUPT ENABLE BIT
115536  005037  002722                  CLR     LKSFL                           ;CLEAR INTERRUPTS FLAG
115542  012702  077777                  MOV     #77777,R2                       ;COUNTER TO WAIT FOR INTERRUPTS
115546  106427  000240                  MTPS    #240                            ;LOWER PRIORITY TO 5
115552  022737  000003  002722  2$:     CMP     #3,LKSFL                        ;3 INTERRUPTS HAPPENED?
115560  001401                  BEQ     3$                              ;EXIT LOOP, IF SO
115562  077205                  SOB     R2,2$                           ;OTHERWISE, KEEP WAITING
115564  106427  000340          3$:     MTPS    #340                            ;RAISE PRIORITY TO 7
115570  000005                  RESET                                   ;EXECUTE RESET
115572  032737  000200  177546  BIT     #BIT07,LKS                      ;READY BIT SET?
115600  001001                  BNE     4$                              ;IF SO, BRANCH
115602  104070                  ERROR   +70                             ;RESET DOESN'T SET LKS<07.
115604  032737  000100  177546  4$:     BIT     #BIT06,LKS                      ;INTERRUPT ENABLE BIT CLEARED?
115612  001401                  BEQ     TST42           ;;IF SO, EXIT TEST
115614  104071                  ERROR   +71                             ;RESET DOESN'T CLEAR LKS
```

```
  19491                                 .SBTTL  TEST  - LINE CLOCK INTERRUPTS
  19492                                 ;LINE CLOCK INTERRUPTS(*)
  19493                                 ;BY SETTING TO 1 LKS<06>, THIS TEST WILL CHECK FOR INTERRUPTS
  19494                                 ;FROM BEVENT LINE AND FROM KDJ11-B 50HZ, 60HZ, 800HZ ON BOARD SIGNALS
  19495                                 ;(THE LATTER SIGNALS WILL BE ACCESSED BY SETTING BCSR<11-10>).
  19496                                 ;
  19497                                 ;BCSR   <11>    <10>            CLOCK SELECT BITS 1 AND 0
  19498                                 ;
  19499                                 ;        0       0              EXTERNAL BEVENT LINE
  19500                                 ;        0       1              ON-BOARD 50 HZ
  19501                                 ;        1       0              ON-BOARD 60 HZ
  19502                                 ;        1       1              ON-BOARD 800 HZ
  19503                                 ;
  19504                                 ;ROUTINE TEST
  19505                                 ;IF UFD THEN
  19506                                 ;.      IF LKS DISABLED THEN
  19507                                 ;.      .       EXIT TEST
  19508                                 ;.      ENDIF
  19509                                 ;.      SET FLAGS TO RUN ONLY WHAT SPECIFIED IN EAPROM
  19510                                 ;ENDIF
  19511                                 ;.      LET 100=ADDRESS OF LKS_INTERRUPT
  19512                                 ;.      DO FOR BCSR<11;10> FROM #0 TO #3
  19513                                 ;.      .       LET LKS<06>=#1
  19514                                 ;.      .       WAIT FOR 10 INTERRUPTS FOR EACH CLOCK
  19515                                 ;.      .       STORE ACTUAL NUMBER OF INTERRUPTS FOR EACH CLOCK
  19516                                 ;.      .       LET INTERRUPT_FLAG=0
  19517                                 ;.      ENDDO
  19518                                 ;.      COMPARE NUMBER OF INTERRUPTS FOR EACH CLOCK
  19519                                 ;ENDROUTINE
  19520                                 ;
  19521                                 ;ROUTINE LKS_INTERRUPT
  19522                                 ;.      INCREMENT INTERRUPT_FLAG
  19523                                 ;RETURN
  19524
  19525                                 ;;*****************************************************************
      115616  000004                    TST42:  SCOPE
  19526 115620  000240                          NOP                             ; THIS CODE ADDED FOR APT DEFAULT BREAK
  19527                                                                         ; PURPOSES-- TEST TIME LONGER THAN SOME APT BREAK IN
TERVALS !
  19528 115622  005737  003032                  TST     CCMPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
  19529 115626  001003                          BNE     99$                     ; NOT YET
  19530 115630  000240                          NOP                             ; DEBUG AID
  19531 115632  000137  116100                  JMP     10$                     ; YES SKIP THIS
  19532 115636  000240                  99$:    NOP
  19533 115640  032737  000100  000052          BIT     #BIT06,@#52                     ;UFD MODE?
  19534 115646  001411                          BEQ     1$                      ;IF NOT, GO DO THE TEST
  19535 115650  032737  010000  177520          BIT     #BIT12,BCSR             ;LKS IS DISABLED?
  19536 115656  001123                          BNE     TST43           ;;IF DISABLED, EXIT TESTS
  19537 115660  005002                          CLR     R2                      ;CLEAR R2 TO SET FLAGS
  19538 115662  053702  177520                  BIS     BCSR,R2                 ;SET R2 ACCORDING TO BCSR
  19539 115666  042702  171777                  BIC     #171777,R2              ;LEAVE ONLY BITS <11-10>
  19540                                 ;
  19541                                 ; SETUP DELAY VALUES FOR INTERRUPTS, IN UFD MODE ONLY FROM THE CLOCK SPECIFIED IN BCSR<11-10
>
  19542                                 ;
  19543 115672  013737  177520  002730  1$:     MOV     BCSR,SAVBR              ;STORE BCSR
  19544 115700  012737  134376  000100          MOV     #LKSINT,100             ;SET UP LKS VECTOR
  19545 115706  012737  000340  000102          MOV     #340,102                ;AT PRIORITY 7
  19546 115714  012705  116116                  MOV     #TIMDEL,R5              ;POINTER TO DEL
```

TEST - LINE CLOCK INTERRUPTS

```
19547 115720  012704  000004                    MOV     #4,R4                   ;R4 IS THE COUNTER FOR ALL CLOCKS
19548 115724  005037  177520                    CLR     BCSR                    ;DO FOR BEVENT LINE INTERRUPTS
19549 115730  000403                            BR      3$                      ;GO DO THE LOOP
19550 115732  062737  002000  177520  2$:       ADD     #2000,BCSR              ;SET UP FOR THE NEXT CLOCK LINE
19551 115740  032737  000100  000052  3$:       BIT     #BIT06,@#52             ;UFD MODE?
19552 115746  001402                            BEQ     4$                      ;IF NOT, BRANCH
19553 115750  010237  177520                    MOV     R2,BCSR                 ;IN UFD, DO ONLY FOR SPECIFIED
19554 115754  005037  002722            4$:     CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
19555 115760  052737  000100  177546            BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19556 115766  012703  000010                    MOV     #10,R3                  ;START COUNTER FOR 10 INTERURRUPTS
19557 115772  012701  177777            5$:     MOV     #177777,R1              ;START COUNTER TO WAIT FOR INTERRUPT
19558 115776  106427  000240                    MTPS    #240                    ;LOWER PRIORITY TO 5
19559 116002  023703  002722            6$:     CMP     LKSFL,R3                ;NEW INTERRUPT HAPPENED?
19560 116006  001401                            BEQ     7$                      ;IF SO, EXIT WAIT LOOP
19561 116010  077104                            SOB     R1,6$                   ;OTHERWISE, KEEP WAITING
19562 116012  010125            7$:     MOV     R1,(R5)+                ;STORE DELAY FOR EACH CLOCK
19563 116014  032737  000100  000052            BIT     #BIT06,@#52             ;UFD MODE?
19564 116022  001026                            BNE     10$                     ;IF UFD, DON'T DO FOR ANY OTHER
19565 116024  077436                            SOB     R4,2$                   ;ALL LINE CLOCKS DONE?
19566                                 ;
19567                                 ; CHECK THE DELAY VALUES FOR ALL CLOCKS
19568                                 ;
19569 116026  106427  000340                    MTPS    #340                    ;RAISE PRIORITY
19570 116032  042737  000100  177546            BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
19571 116040  012705  116116                    MOV     #TIMDEL,R5              ;POINTER TO DELAY TABLE
19572 116044  021565  000006                    CMP     (R5),6(R5)              ;DELAY FOR BEVENT AND 800HZ?
19573 116050  103401                            BLO     8$                      ;BEVENT IS NOT 800HZ
19574 116052  104064                            ERROR   +64                     ;WRONG # OF INTERRUPTS
19575 116054  005725            8$:     TST     (R5)+                   ;INCREMENT POINTER
19576 116056  021565  000002                    CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 60HZ?
19577 116062  103401                            BLO     9$                      ;IF FIRST BIGGER, BRANCH
19578 116064  104064                            ERROR   +64                     ;WRONG # OF INTERRUPTS
19579 116066  005725            9$:     TST     (R5)+                   ;INCREMENT POINTER
19580 116070  021565  000002                    CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 800HZ
19581 116074  103401                            BLO     10$                     ;IF FIRST BIGGER, BRANCH
19582 116076  104064                            ERROR   +64                     ;WRONG # OF INTERRUPTS
19583 116100  013737  002730  177520  10$:      MOV     SAVBR,BCSR              ;RESTORE BCSR
19584 116106  042737  000100  177546            BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
19585 116114  000404                            BR      TST43           ;;EXIT TEST
19586
19587 116116                          TIMDEL: .BLKW 4
19588
```

```
19590                                   .SBTTL  TEST  - MAINTENANCE REGISTER TEST
19591                                   ;MAINTENANCE REGISTER TEST
19592                                   ;THIS TEST WILL ADDRESS MAINTENANCE REGISTER AND CHECK BITS
19593                                   ;7-4 TO BE 0010, 2-1 TO BE 10, AND READ BITS 10-08, 03, 00
19594                                   ;FOR FUTURE USE.  THOSE BITS REPRESENT THE FOLLOWING SIGNALS:
19595                                   ;MULTIPROCESSOR SLAVE, UNIBUS SYSTEM, FPA AVAILABLE, HALT/TRAP
19596                                   ;OPTION, AND AC POWER OKAY.
19597                                   ;ROUTINE TEST
19598                                   ;.      IF MAINT. REG. BITS <7-4> NE 0010 OR <2-1> NE 10 THEN
19599                                   ;.          ERROR
19600                                   ;.      ENDIF
19601                                   ;.      READ MAINT.REG. BITS <10-08,03,00>
19602                                   ;ENDROUTINE
19603
19604                                   ;;**********************************************************************
     116126  000004                     TST43:  SCOPE
19605 116130  032737  174000  177750            BIT     #174000,MAIREG                  ;UNUSED BITS ALL ZEROS?
19606 116136  001401                            BEQ     1$                              ;IF OK, BRANCH
19607 116140  104132                            ERROR   .132                            ;MAINTENANCE REGISTER ERROR
19608 116142  032737  000044  177750  1$:       BIT     #44,MAIREG                      ;<5,2> SET ?
19609 116150  001001                            BNE     2$                              ;IF SO, BRANCH
19610 116152  104132                            ERROR   .132                            ;MAINTENANCE REGISTER ERROR
19611 116154  032737  000322  177750  2$:       BIT     #322,MAIREG                     ;<7,6,4,1> CLEAR?
19612 116162  001401                            BEQ     TST44           ;;IF YES, BRANCH
19613 116164  104132                            ERROR   .132
19614
19615
```

```
                                        .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

                                        ;;*****************************************************************
     116166  000004                     TST44:  SCOPE
     116170  032737  000100  000052             BIT     #BIT06,B#52                     ;UFD MODE?
     116176  001406                             BEQ     1$                              ;IF NOT, GO DO THE TEST
     116200  032737  000200  177524             BIT     #BIT07,BCR                      ;IF UFD AND CONSOLE NOT PRESENT
     116206  001402                             BEQ     1$                              ;NOT TRUE, DO THE TEST
     116210  000137  120134                     JMP     SLEND                           ;IF TRUE, SKIP ALL SLU TESTS
                                        ;
                                        ; TRY TO ACCESS SLU REGISTERS
                                        ;
     116214  013701  000004             1$:     MOV     ERRVEC,R1                       ;SAVE TIMEOUT VECTOR
     116220  012737  116244  000004             MOV     #3$,ERRVEC                      ;POINT NEW ONE TO PROGRAM AREA
     116226  012737  000340  000006             MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
     116234  012702  177560                     MOV     #RCSR,R2                        ;START ACCESSING WITH RCSR
     116240  005712                     2$:     TST     (R2)                            ;ACCESS SLU REGISTER
     116242  000403                             BR      4$                              ;IF NO TIMEOUT, CONTINUE
     116244  010237  001126             3$:     MOV     R2,$BDDAT                       ;STORE ADDRESS THAT TIMED OUT
     116250  104072                             ERROR   +72                             ;TIMEOUT ACCESSING SLU REGISTER
     116252  022722  177566             4$:     CMP     #XBUF,(R2)+                     ;LAST REGISTER ACCESSED?
     116256  103770                             BLO     2$                              ;IF NOT, BRANCH
     116260  010137  000004                     MOV     R1,ERRVEC                       ;RESTORE TIMEOUT VECTOR
```

TEST - XCSR BIT 7

```
                                .SBTTL  TEST  - XCSR BIT 7
                                ;CHECK THAT XCSR<07> CAN BE 0 AND 1.
                                ;
                                ;XCSR   <07>            TRANSMITTER READY
                                ;
                                ;ROUTINE TEST
                                ;.      WAIT FOR XCSR<07>=#1 NO MORE THAN 200MSEC
                                ;.      IF XCSR<07> NE #1 THEN
                                ;.          .       ERROR
                                ;.      ENDIF
                                ;.      LET XBUF=#NULL
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#NULL
                                ;.      IF XCSR<07> NE 0 THEN
                                ;.          .       ERROR (READY DIDN'T GO LOW)
                                ;.      ENDIF
                                ;ENDROUTINE
                                ;;**********************************************************************
116264  000004                TST45:  SCOPE
116266  012701  001000                MOV     #1000,R1                ;COUNTER FOR ABOUT 200MICROSEC.
116272  122737  000001  001220        CMPB    #APTENV,$ENV            ;RUNNING IN APT MODE?
116300  001003                        BNE     1$                      ;NO, GO DO TEST
116302  005737  001206                TST     $PASS                   ;FIRST PASS?
116306  001017                        BNE     TST46           ;;IF APT AND NOT FIRST PASS, EXIT TEST
116310  105737  177564        1$:     TSTB    XCSR                    ;XCSR<7> READY 1?
116314  100401                        BMI     2$                      ;IF SO, EXIT WAIT LOOP
116316  077104                        SOB     R1,1$                   ;IF NOT 1, CONTINUE WAITING
116320  105737  177564        2$:     TSTB    XCSR                    ;XCSR<7>=1?
116324  100401                        BMI     3$                      ;IF YES, BRANCH
116326  104073                        ERROR   .73                     ;XCSR<7> DOES NOT BECOME 1
116330  012737  000000  177566  3$:   MOV     #NULL,XBUF              ;TRY TO TRANSMIT NULL CHARACTER
116336  105737  177564                TSTB    XCSR                    ;XCSR<7>=0
116342  100001                        BPL     TST46           ;;IF YES, EXIT TEST
116344  104073                        ERROR   .73                     ;XMIT READY DIDN'T GO LOW
```

TEST - RCSR BIT 7 AND XCSR BIT 2

```
                                .SBTTL  TEST  - RCSR BIT 7 AND XCSR BIT 2
                                ;CHECK THAT RCSR<07> CAN BE 0 AND 1 AND THAT XCSR<02> WORKS PROPERLY.
                                ;
                                ;RCSR   <07>    RECEIVER DONE
                                ;XCSR   <02>    MAINTENANCE
                                ;
                                ;ROUTINE TEST
                                ;.(CHECK RCSR<07> AND XCSR<07>)
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XCSR<02>=#1 (LOOP BACK MODE)
                                ;.      LET XBUF=#125
                                ;.      WAIT FOR RCSR<07>=#1 NO MORE THAN 200MSEC
                                ;.      IF RCSR<07> NE #1 THEN
                                ;.      .       ERROR (RCSR<07> DOES NOT BECOME 1 OR XCSR<02>DOES NOT
                                ;.      .               WORK)
                                ;.      ENDIF
                                ;.      IF RBUF NE #125 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      IF RCSR<07> NE #0 THEN
                                ;.      .       ERROR (RCSR<07>DOES NOT GO LOW)
                                ;.      ENDIF
                                ;.      LET XCSR<02>=#0
                                ;ENDROUITNE

                                ;;*****************************************************************
116346  000004                  TST46:  SCOPE
19730 116350  012701  000013          MOV     #13,R1                  ;COUNTER FOR ABOUT 200MICROSEC.
19731 116354  122737  000001  001220  CMPB    #APTENV,$ENV            ;RUNNING IN APT MODE?
19732 116362  001003                  BNE     1$                      ;NO, GO DO TEST
19733 116364  005737  001206          TST     $PASS                   ;FIRST PASS?
19734 116370  001074                  BNE     TST47           ;;IF APT AND NOT FIRST PASS, EXIT TEST
19735 116372  105737  177564  1$:     TSTB    XCSR                    ;XCSR<7> READY 1?
19736 116376  100375                  BPL     1$                      ;IF NOT 1, CONTINUE WAITING
19737 116400  052737  000004  177564  2$:     BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
19738 116406  032737  000004  177564          BIT     #BIT02,XCSR             ;GOT SET OK?
19739 116414  001004                  BNE     3$                      ;IF YES, BRANCH
19740 116416  005037  177564          CLR     XCSR                    ;RESET TO PRINT ERROR
19741 116422  104114                  ERROR   +114                    ;XCSR<2> DOES NOT BECOME 1
19742 116424  000456                  BR      TST47           ;;EXIT TEST
19743                                 ;
19744                                 ; STALL FOR A WHILE IN CASE XCSR<2> CAUSES RCSR<7> TO BE 1
19745                                 ;
19746 116426  012701  060000  3$:     MOV     #60000,R1               ;STALL IN CASE XCSR<2> SETS READY
19747 116432  105737  177560  4$:     TSTB    RCSR                    ;IF RECEIVER READY SET?
19748 116436  100402                  BMI     5$                      ;IF SET, BRANCH
19749 116440  077104                  SOB     R1,4$                   ;OTHERWISE, STAY FOR A WHILE
19750 116442  000402                  BR      6$                      ;IF NOT READY, BRANCH
19751 116444  005737  177562  5$:     TST     RBUF                    ;READ RBUF
19752                                 ;
19753                                 ; TRANSMIT XON AND CHECK RCSR<7>
19754                                 ;
19755 116450  012737  000021  177566  6$:     MOV     #21,XBUF                ;TRANSMIT A CHARACTER
19756 116456  012701  060000          MOV     #60000,R1               ;COUNTER TO WAIT
19757 116462  105737  177560  7$:     TSTB    RCSR                    ;RCSR<7> READY 1?
19758 116466  100401                  BMI     8$                      ;IF YES, EXIT WAIT LOOP
19759 116470  077104                  SOB     R1,7$                   ;OTHERWISE, CONTINUE WAITING
```

TEST - RCSR BIT 7 AND XCSR BIT 2

```
19760 116472  105737  177560          8$:     TSTB    RCSR                ;RCSR<7>=1?
19761 116476  100403                          BMI     9$                  ;IF YES, BRANCH
19762 116500  005037  177564                  CLR     XCSR                ;RESET XCSR<2>
19763 116504  104074                          ERROR   +74                 ;RECEIVER READY DIDN'T COME UP
19764 116506  013737  177562  001126  9$:     MOV     RBUF,$BDDAT         ;STORE RECEIVED DATA
19765 116514  022737  000021  001126          CMP     #21,$BDDAT          ;DATA RECEIVED OK?
19766 116522  001406                          BEQ     10$                 ;IF YES, BRANCH
19767 116524  012737  000021  001124          MOV     #21,$GDDAT
19768 116532  005037  177564                  CLR     XCSR                ;RESET TO ENABLE SLU
19769 116536  104075                          ERROR   +75                 ;WRONG CHARACTER RECEIVED
19770 116540  105737  177560          10$:    TSTB    RCSR                ;RCSR<7>=0?
19771 116544  100003                          BPL     11$                 ;IF ZERO, BRANCH
19772 116546  005037  177564                  CLR     XCSR                ;RESET TO ENABLE SLU
19773 116552  104076                          ERROR   +76                 ;RCSR<07><>0 AFTER READING RBUF
19774 116554  042737  000004  177564  11$:    BIC     #BIT02,XCSR         ;DISABLE LOOP BACK MODE
19775
19776
```

TEST - RESET AND XCSR<2!0>

```
 19778                                  .SBTTL  TEST  - RESET AND XCSR<2!0>
 19779                                  ;CHECK THAT RESET CLEARS XCSR<0!2>.
 19780                                  ;ROUTINE TEST
 19781                                  ;.(CHECK RCSR<07> AND XCSR<07> AND RESET)
 19782                                  ;.      LET XCSR<02,00>=#1 (LOOP BACK MODE)
 19783                                  ;.      EXECUTE "RESET"
 19784                                  ;.      IF XCSR<02!00> NE #0 THEN
 19785                                  ;.              ERROR
 19786                                  ;.      ENDIF
 19787                                  ;.      LET XCSR<02>=#0
 19788                                  ;ENDROUTINE
 19789
 19790                                  ;;*****************************************************************
      116562  000004                    TST47:  SCOPE
 19791 116564  005737  001206                    TST     $PASS                           ;FIRST PASS?
 19792 116570  001011                    BNE     TST50           ;;IF NOT FIRST PASS, EXIT TEST
 19793 116572  052737  000005  177564  1$:     BIS     #BIT02!BIT00,XCSR               ;LOOP BACK MODE
 19794                                  ;
 19795                                  ; EXECUTE RESET AND VALIDATE THAT XCSR<7,2> BECOMES <1,0>
 19796                                  ;
 19797 116600  000005                    RESET                                           ;EXECUTE RESET
 19798 116602  032737  000005  177564    BIT     #BIT02!BIT00,XCSR               ;XCSR<2,0> CLEAR?
 19799 116610  001401                    BEQ     TST50           ;;IF YES, BRANCH
 19800 116612  104102                    ERROR   +102                            ;XCSR<2,0> NOT CLEARED ON RESET
 19801
 19802
```

```
                                .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

                                ;;*****************************************************************
116614  000004                  TST50:  SCOPE
19833 116616  005737  001206            TST     $PASS                   ;FIRST PASS?
19834 116622  001033            BNE     5$                              ;SKIP RESET PART OF THE TEST
19835                           ;
19836                           ; CHECK THAT INTERRUPTS ENABLE BITS FOR RECEIVER AND TRASMITTER OF SLU
19837                           ; ARE CLEARED BY RESET
19838                           ;
19839 116624  052737  000100  177564  1$:     BIS     #BIT06,XCSR             ;SET INTERRUPT ENABLE BIT IN XCSR
19840 116632  032737  000100  177564          BIT     #BIT06,XCSR             ;GOT SET OK?
19841 116640  001001                        BNE     2$                      ;IF YES, BRANCH
19842 116642  104110                        ERROR   .110                    ;IN BIT 6 OF XCSR
19843 116644  052737  000100  177560  2$:     BIS     #BIT06,RCSR             ;SET INTERRUPT ENABLE BIT IN RCSR
19844 116652  032737  000100  177560          BIT     #BIT06,RCSR             ;GOT SET OK?
19845 116660  001001                        BNE     3$                      ;IF YES, BRANCH
19846 116662  104110                        ERROR   .110                    ;IN BIT 6 OF RSCR
19847 116664  000005            3$:     RESET                           ;INLINE BUS RESET
19848 116666  032737  000100  177564          BIT     #BIT06,XCSR             ;XMIT INTERRUPT ENABLE BIT CLEARED?
19849 116674  001401                        BEQ     4$                      ;IF CLEARED, BRANCH
19850 116676  104102                        ERROR   .102                    ;INTERRUPT ENABLE NOT CLEARED ON RESET
19851 116700  032737  000100  177560  4$:     BIT     #BIT06,RCSR             ;RECEIVE INTERRUPT ENBLE CLEARED?
19852 116706  001401                        BEQ     5$                      ;IF CLEARED, BRANCH
19853 116710  104102                        ERROR   .102                    ;INTERRUPT ENABLE NOT CLEARED ON RESET
19854                           ;
19855                           ; CHECK THAT TRANSMIT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19856                           ;
19857 116712  012737  116760  000064  5$:     MOV     #9$,@#64                ;POINT XMIT VECTOR TO PROGRAM AREA
19858 116720  012737  000340  000066          MOV     #340,@#66               ;AT PRIORITY 7
19859 116726  052737  000100  177564          BIS     #BIT06,XCSR             ;SET INTERRUPT ENABLE BIT IN XCSR
```

TEST - RESET AND INTERRUPT ENABLE BITS

```
19860 116734  012702  000340                    MOV     #340,R2                 ;SET PRIORITY TO 7
19861 116740  000402                            BR      7$                      ;GO WAIT IN CASE OF INTERRUPTS
19862 116742  162702  000040            6$:     SUB     #40,R2                  ;LOWER PRIORITY LEVEL
19863 116746  106402                    7$:     MTPS    R2                      ;SET PRIORITY
19864 116750  012703  000026                    MOV     #26,R3                  ;TIME DELAY
19865 116754  077301                    8$:     SOB     R3,8$                   ;WAIT FOR INTERRUPTS
19866 116756  000403                            BR      10$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19867 116760  104101                    9$:     ERROR   +101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
19868 116762  005726                            TST     (SP)+                   ;CLEAN UP THE STACK
19869 116764  005726                            TST     (SP)+
19870 116766  022702  000200            10$:    CMP     #200,R2                 ;AT PRIORITY 4?
19871 116772  001363                            BNE     6$                      ;IF NOT LAST ONE, CONTINUE
19872 116774  106427  000340                    MTPS    #340                    ;RESTORE PRIORITY 7
19873 117000  042737  000100  177564            BIC     #BIT06,XCSR             ;CLEAR INTERRUPT ENABLE BIT
19874                                           ;
19875                                           ; CHECK THAT RECEIVE INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19876                                           ;
19877 117006  012737  117076  000060            MOV     #15$,@#60               ;POINT RECEIVE VECTOR TO PROGRAM AREA
19878 117014  012737  000340  000062            MOV     #340,@#62               ;AT PRIORITY 7
19879 117022  105737  177564            11$:    TSTB    XCSR                    ;TRANSMITTER READY
19880 117026  100375                            BPL     11$                     ;IF NOT, WAIT
19881 117030  012737  000000  177566            MOV     #NULL,XBUF              ;TRY TO TRANSMIT NULL
19882 117036  052737  000100  177560            BIS     #BIT06,RCSR             ;SET INTERRUPT ENABLE BIT IN RCSR
19883 117044  012702  000340                    MOV     #340,R2                 ;SET PRIORITY TO 7
19884 117050  000402                            BR      13$                     ;GO WAIT IN CASE OF INTERRUPTS
19885 117052  162702  000040            12$:    SUB     #40,R2                  ;LOWER PRIORITY LEVEL
19886 117056  052737  000004  177564    13$:    BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
19887 117064  106402                            MTPS    R2                      ;SET PRIORITY
19888 117066  012703  000100                    MOV     #100,R3                 ;TIME DELAY
19889 117072  077301                    14$:    SOB     R3,14$                  ;WAIT FOR INTERRUPTS
19890 117074  000406                            BR      16$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19891 117076  042737  000004  177564    15$:    BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
19892 117104  104101                            ERROR   +101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
19893 117106  005726                            TST     (SP)+                   ;CLEAN UP THE STACK
19894 117110  005726                            TST     (SP)+
19895 117112  022702  000200            16$:    CMP     #200,R2                 ;AT PRIORITY 4?
19896 117116  001355                            BNE     12$                     ;IF NOT LAST ONE, CONTINUE
19897                                           ;
19898                                           ; CLEAN UP BEFORE NEXT TEST
19899                                           ;
19900 117120  106427  000340                    MTPS    #340                    ;RESTORE PRIORITY 7
19901 117124  042737  000100  177560            BIC     #BIT06,RCSR             ;CLEAR INTERRUPT ENABLE BIT
19902 117132  012702  000300                    MOV     #300,R2                 ;STALL DELAY
19903 117136  105737  177560            17$:    TSTB    RCSR                    ;RECEIVE READY?
19904 117142  100401                            BMI     18$                     ;STOP WAITING, IF SO
19905 117144  077204                            SOB     R2,17$                  ;OTHERWISE, STAY IN THE LOOP
19906 117146  005737  177562            18$:    TST     RBUF                    ;READ CHARACTER TRANSMITTED
19907 117152  042737  000004  177564            BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
19908
```

```
19910                                   .SBTTL  TEST  - INTERRUPT PRIORITY FOR SLU
19911                                   ;CHECK THAT INTERRUPTS HAPPEN AT PRIORITY 3 AND THAT THEY CLEAR
19912                                   ;RCSR<06> AND XCSR<06>.
19913                                   ;
19914                                   ;ROUTINE TEST
19915                                   ;.      LET 60=@ADDRESS_OF_LEGAL_RINTERRUPT
19916                                   ;.      LET 64=@ADDRESS_OF_LEGAL_XINTERRUPT
19917                                   ;.      LET XCSR<02>=#1
19918                                   ;.      SET PRIORITY TO #3
19919                                   ;.      WAIT FOR XINTERRUPT=#3
19920                                   ;.      IF XCSR<07> EQ #1 THEN
19921                                   ;.          .       ERROR
19922                                   ;.      ENDIF
19923                                   ;.      WAIT FOR RINTERRUPT=#3
19924                                   ;.      IF RCSR<07> EQ #0 THEN
19925                                   ;.          .       ERROR
19926                                   ;.      ENDIF
19927                                   ;.      LET XCSR<02>=#0
19928                                   ;.      SET PRIORITY TO NORMAL
19929                                   ;ENDROUTINE
19930                                   ;
19931                                   ;ROUTINE LEGAL_XINTERRUPT
19932                                   ;.      LET XBUF=@CHARACTER
19933                                   ;.      INCREMENT XINTERRUPT
19934                                   ;ENDROUTINE
19935                                   ;
19936                                   ;ROUTINE LEGAL_RINTERRUPT
19937                                   ;.      READ RCSR
19938                                   ;.      INCREMENT RINTERRUPT
19939                                   ;ENDROUTINE
19940
19941                                   ;;*****************************************************************************
     117160  000004                     TST51:  SCOPE
19942 117162  122737  000001  001220             CMPB    @APTENV,$ENV                    ;RUNNING IN APT MODE?
19943 117170  001003                             BNE     100$                            ;NO, GO DO TEST
19944 117172  005737  001206                     TST     $PASS                           ;FIRST PASS?
19945 117176  001113                             BNE     TST52           ;;IF APT AND NOT FIRST PASS, EXIT TEST
19946                                   ;
19947                                   ; GET READY FOR INTERRUPTS
19948                                   ;
19949 117200  012737  117374  000060  100$:    MOV     #8$,@#60                        ;STORE RECEIVER VECTOR
19950 117206  012737  117320  000064             MOV     #6$,@#64                        ;STORE TRANSMITTER VECTOR
19951 117214  012737  000340  000062             MOV     #340,@#62                       ;AT PRIORITY 7
19952 117222  012737  000340  000066             MOV     #340,@#66                       ;FOR RECEIVER AND TRANSMITTER
19953 117230  052737  000004  177564             BIS     #BIT02,XCSR                     ;SET LOOP BACK MODE
19954 117236  012701  000100                     MOV     #100,R1                         ;DELAY FOR UNEXPECTED CHARACTERS
19955 117242  105737  177560             1$:     TSTB    RCSR                            ;RECEIVER READY?
19956 117246  100401                             BMI     2$                              ;IF YES, BRANCH
19957 117250  077104                             SOB     R1,1$                           ;OTHERWISE, WAIT JUST IN CASE
19958 117252  005737  177562             2$:     TST     RBUF                            ;READ RECEIVER
19959                                   ;
19960                                   ; SET PRIORITIES AND XMIT INTERRUPTS
19961                                   ;
19962 117256  012702  000140                     MOV     #140,R2                         ;START WITH PRIORITY 3
19963 117262  000402                             BR      4$                              ;TRY TO DO IT
19964 117264  162702  000040             3$:     SUB     #40,R2                          ;LOWER PRIORITY
19965 117270  106402                     4$:     MTPS    R2                              ;TRY TO DO AT LOWER PRIORITY
```

TEST - INTERRUPT PRIORITY FOR SLU

```
19966 117272  052737  000100  177564          BIS     #BIT06,XCSR                     ;LOOP BACK & INTERRUPT ENABLE
19967 117300  012703  001000                  MOV     #1000,R3                        ;WAIT DELAY FOR INTERRUPTS
19968 117304  077301                  5$:     SOB     R3,5$                           ;WAIT FOR XMIT INTERRUPTS
19969 117306  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK BIT
19970 117314  104107                          ERROR   +107                            ;NO XMIT INTERRUPTS
19971 117316  000443                          BR      TST52           ;;IF ERROR, EXIT TEST
19972                                 ;
19973                                 ; TRANSMITTER INTERRUPT HERE
19974                                 ;
19975 117320  005726                  6$:     TST     (SP)+                           ;CLEAN UP STACK
19976 117322  005726                          TST     (SP)+
19977 117324  042737  000100  177564          BIC     #BIT06,XCSR                     ;CLEAR INTERRUPT ENABLE
19978 117332  012737  000000  177566          MOV     #NULL,XBUF                      ;TRANSMIT NULL
19979 117340  052737  000100  177560          BIS     #BIT06,RCSR                     ;SET RECEIVE INTERRUPT
19980 117346  106402                          MTPS    R2                              ;SET NEXT PRIORITY
19981 117350  012703  100000                  MOV     #100000,R3                      ;WAIT DELAY FOR INTERRUPTS
19982 117354  077301                  7$:     SOB     R3,7$                           ;WAIT FOR RECEIVE INTERUPTS
19983 117356  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK MODE BIT
19984 117364  104107                          ERROR   +107                            ;NO RECEIVE INTERRUPTS
19985 117366  000406                          BR      9$                              ;DON'T TOUCH STACK
19986 117370  106427  000340                  MTPS    #340                            ;RAISE PRIORITY
19987                                 ;
19988                                 ; RECEIVER INTERRUPT HERE
19989                                 ;
19990 117374  005726                  8$:     TST     (SP)+                           ;CLEAN UP STACK
19991 117376  005726                          TST     (SP)+
19992 117400  005737  177562                  TST     RBUF                            ;READ RECEIVER BUFFER
19993 117404  005702                  9$:     TST     R2                              ;PRIORITY 0
19994 117406  001326                          BNE     3$                              ;IF NOT YET, CONTINUE
19995 117410  106427  000340                  MTPS    #340                            ;RAISE PRIORITY TO 7
19996 117414  042737  000100  177560          BIC     #BIT06,RCSR                     ;CLEAR RECEIVE INTER. ENABLE
19997 117422  005037  177564                  CLR     XCSR                            ;CLEAR XCSR
19998
```

TEST - BREAK CONDITION

```
20000                                   .SBTTL  TEST  - BREAK CONDITION
20001                                   ;CHECK THAT SENDING BREAK CAUSES FRAMING ERROR.
20002                                   ;
20003                                   ;RCSR   <15>    ERROR
20004                                   ;       <13>    FRAMING ERROR
20005                                   ;       <11>    RECEIVED BREAK
20006                                   ;
20007                                   ;XCSR   <00>    TRANSMIT BREAK
20008                                   ;
20009                                   ;ROUTINE TEST
20010                                   ;.      LET XCSR<02>=#1
20011                                   ;.      LET XCSR<00>=#1
20012                                   ;.      WAIT FOR RCSR<07>=#1
20013                                   ;.      IF RBUF<15!13!11> NE #1 THEN
20014                                   ;.      .       ERROR (ERROR, FRAMING ERROR, RECEIVE BREAK NE 1)
20015                                   ;.      ENDIF
20016                                   ;.      LET XCSR<00>=#0
20017                                   ;.      IF XCSR<00> NE #0 THEN
20018                                   ;.      .       ERROR (XCSR<00> DOES NOT GO LOW)
20019                                   ;.      ENDIF
20020                                   ;.      WAIT FOR XCSR<07>=#1
20021                                   ;.      LET XBUF=#NULL (SEND NULL CHARACTER TO SEE ERROR CLEARED)
20022                                   ;.      WAIT FOR RCSR<07>=#1
20023                                   ;.      IF RBUF<15!13!11> NE #0 THEN
20024                                   ;.      .       ERROR
20025                                   ;.      ENDIF
20026                                   ;.      LET XCSR<00>=#1
20027                                   ;.      EXECUTE "RESET"
20028                                   ;.      IF XCSR<00> NE #0 THEN
20029                                   ;.      .       ERROR
20030                                   ;.      ENDIF
20031                                   ;.      LET XCSR<02>=#0
20032                                   ;ENDROUTINE
20033
20034                                   ;;*****************************************************************************
    117426  000004                      TST52:  SCOPE
20035                                   ;
20036                                   ; DECIDE WHETHER TO RUN THIS TEST
20037                                   ;
20038 117430  032737  000200  000052            BIT     #BIT07,@#52                     ;UFD MODE?
20039 117436  001127                            BNE     TST53           ;;IN UFD MODE, EXIT TEST
20040 117440  005737  001206                    TST     $PASS                           ;FIRST PASS?
20041 117444  001124                            BNE     TST53           ;;IF APT AND NOT FIRST PASS, EXIT TEST
20042                                   ;
20043                                   ; SEND BREAK AND CHECK ERROR BITS IN RBUF
20044                                   ;
20045 117446  052737  000004  177564    1$:     BIS     #BIT02,XCSR                     ;TRANSMIT IN LOOP BACK
20046 117454  013737  177520  002730            MOV     BCSR,SAVBR                      ;SAVE BCSR
20047 117462  042737  001000  177520            BIC     #BIT09,BCSR                     ;DISABLE HALT ON BREAK
20048 117470  052737  000001  177564            BIS     #BIT00,XCSR                     ;SET SEND BREAK BIT
20049 117476  032737  000001  177564            BIT     #BIT00,XCSR                     ;GOT SET OK?
20050 117504  001001                            BNE     2$                              ;IF YES, BRANCH
20051 117506  104110                            ERROR   +110                            ;WRITING 1 TO XCSR<0>
20052 117510  012701  000100            2$:     MOV     #100,R1                         ;STALL DELAY
20053 117514  105737  177560            4$:     TSTB    RCSR                            ;RECEIVER READY?
20054 117520  100401                            BMI     5$                              ;IF YES, BRANCH
20055 117522  077104                            SOB     R1,4$                           ;WAIT JUST IN CASE OF A CHARACTER
```

TEST - BREAK CONDITION

```
20056 117524  005737  177562          5$:     TST     RBUF                            ;READ A CHARACTER
20057 117530  052737  000001  177564          BIS     #BIT00,XCSR                     ;TRANSMIT BREAK
20058 117536  012701  001000                  MOV     #1000,R1                        ;ANOTHER DELAY TO GET BREAK
20059 117542  077101                  6$:     SOB     R1,6$                           ;WAIT A WHILE
20060 117544  105737  177560          7$:     TSTB    RCSR                            ;RECEIVER READY?
20061 117550  100375                          BPL     7$                              ;IF NOT, WAIT
20062 117552  013737  177562  001126          MOV     RBUF,$BDDAT                     ;STORE WHATEVER RECEIVED
20063 117560  022737  124000  001126          CMP     #BIT15!BIT13!BIT11,$BDDAT       ;ALL ERROR BITS SET?
20064 117566  001405                          BEQ     8$                              ;IF YES, BRANCH
20065 117570  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20066 117576  104105                          ERROR   •105                            ;BREAK DOES NOT CAUSE ERRORS
20067 117600  000446                          BR      TST53           ;;EXIT
20068 117602  042737  000001  177564  8$:     BIC     #BIT00,XCSR                     ;CLEAR TRANSMIT BREAK
20069 117610  032737  000001  177564          BIT     #BIT00,XCSR                     ;GOT CLEARED OK?
20070 117616  001405                          BEQ     9$                              ;IF YES, BRANCH
20071 117620  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20072 117626  104110                          ERROR   •110                            ;ERROR WRITING 0 TO XCSR<0>
20073 117630  000432                          BR      TST53           ;;EXIT
20074                                 ;
20075                                 ; CHECK THAT BREAK CONDITION IS CLEARED
20076                                 ;
20077 117632  013737  002730  177520  9$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
20078 117640  105737  177564          10$:    TSTB    XCSR                            ;XMIT READY?
20079 117644  100375                          BPL     10$                             ;IF NOT, WAIT
20080 117646  012737  000177  177566          MOV     #177,XBUF                       ;TRY TO TRANSMIT DELETE
20081 117654  105737  177560          11$:    TSTB    RCSR                            ;RECEIVER READY
20082 117660  100375                          BPL     11$                             ;IF NOT, WAIT
20083 117662  013737  177562  001126          MOV     RBUF,$BDDAT                     ;STORE RECEIVE BUFFER
20084 117670  032737  124000  001126          BIT     #BIT15!BIT13!BIT11,$BDDAT       ;ERRORS CLEARED?
20085 117676  001404                          BEQ     12$                             ;IF YES, BRANCH
20086 117700  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20087 117706  104106                          ERROR   •106                            ;BREAK NOT CLEARED ON NEXT CHARACTER
20088 117710  042737  000004  177564  12$:    BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK MODE
20089
20090
```

```
                                .SBTTL  TEST  - OVERRUN CONDITION
                                ;CHECK OVERRUN CONDITION
                                ;
                                ;RCSR   <14>    OVERRUN ERROR
                                ;
                                ;ROUTINE TEST
                                ;.      LET XCSR<02>=#1 (LOOPBACK MODE)
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#252
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#125 (SEND THE 2ND W/O READING THE 1ST CHARACTER)
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      STALL FOR LOWEST BAUD RATE TO GET 2ND CHARACTER
                                ;.      IF LOW BYTE OF RBUF NE #125 THEN
                                ;.      .       ERROR (1ST CHARACTER WASN'T OVERRUN)
                                ;.      ENDIF
                                ;.      IF RBUF<15!14> NE #1 THEN
                                ;.      .       ERROR (NO OVERRUN BIT SET)
                                ;.      ENDIF
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#NULL
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      IF RBUF<15!14> NE #0 THEN
                                ;.      .       ERROR (WASN'T CLEARED ON THE NEXT CHARACTER RECEIVED)
                                ;.      ENDIF
                                ;.      LET XCSR<02>=#0
                                ;ENDROUTINE
                                ;
                                ;;*****************************************************************
    117716  000004              TST53:  SCOPE
20121 117720  122737  000001  001220        CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
20122 117726  001003                        BNE     100$                            ;NO, GO DO TEST
20123 117730  005737  001206                TST     $PASS                           ;FIRST PASS?
20124 117734  001077                        BNE     TST54           ;;IF APT AND NOT FIRST PASS, EXIT TEST
20125 117736  052737  000004  177564  100$:  BIS     #BIT02,XCSR                     ;SET LOOP BACK MODE
20126 117744  105737  177564        1$:     TSTB    XCSR                            ;READY TO TRANSMIT?
20127 117750  100375                        BPL     1$                              ;IF NOT, WAIT
20128 117752  012737  000021  177566        MOV     #21,XBUF                        ;TRANSMIT A CHARACTER
20129 117760  105737  177560        2$:     TSTB    RCSR                            ;RECEIVE READY?
20130 117764  100375                        BPL     2$                              ;IF NOT, WAIT
20131 117766  105737  177564        3$:     TSTB    XCSR                            ;READY TO TRANSMIT?
20132 117772  100375                        BPL     3$                              ;IF NOT, WAIT
20133 117774  012737  000177  177566        MOV     #177,XBUF                       ;TRANSMIT THE 2ND CHARACTER
20134 120002  012703  150000                MOV     #150000,R3                      ;STALL FOR THE 2ND CHARACTER
20135 120006  077301                4$:     SOB     R3,4$                           ;WAIT A WHILE
20136 120010  013737  177562  001126        MOV     RBUF,$BDDAT                     ;STORE RECEIVED DATA
20137 120016  012737  140177  001124        MOV     #140177,$GDDAT                  ;EXPETED PATTERN
20138 120024  122737  000177  001126        CMPB    #177,$BDDAT                     ;2ND CHARACTER RECEIVED?
20139 120032  001404                        BEQ     5$                              ;IF YES, BRANCH
20140 120034  042737  000004  177564        BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20141 120042  104111                        ERROR   +111                            ;2ND CHARACTER DIDN'T OVERRUN 1ST
20142 120044  122737  000300  001127  5$:   CMPB    #BIT7!BIT6,$BDDAT+1             ;OVERRUN ERROR BITS SET?
20143 120052  001404                        BEQ     6$                              ;IF YES, BRANCH
20144 120054  005037  177564                CLR     XCSR                            ;RESET TO ENABLE SLU
20145 120060  104112                        ERROR   +112                            ;OVERRUN DOES NOT SET ERR  S BITS
20146 120062  000424                        BR      TST54           ;;EXIT
20147                           ;
```

TEST - OVERRUN CONDITION

```
20148                                   ; SEND NEXT CHARACTER TO CLEAR OVERRUN CONDITIONS
20149                                   ;
20150 120064  105737  177564      6$:   TSTB    XCSR                ;TRANSMITTER READY?
20151 120070  100375                    BPL     6$                  ;IF NOT, BRANCH AND WAIT
20152 120072  012737  000000  177566    MOV     #NULL,XBUF          ;TRANSMIT NULL CHARACTER
20153 120100  105737  177560      7$:   TSTB    RCSR                ;RECEIVER READY?
20154 120104  100375                    BPL     7$                  ;IF NOT, BRANCH AND WAIT
20155 120106  032737  140000  177562    BIT     #BIT15!BIT14,RBUF   ;ANY ERRORS SET?
20156 120114  001404                    BEQ     8$                  ;IF NOT, BRANCH
20157 120116  042737  000004  177564    BIC     #BIT02,XCSR         ;RESET TO ENABLE SLU
20158 120124  104113                    ERROR   .113                ;OVERRUN NOT CLEARED ON NEXT CHAR.
20159 120126  042737  000004  177564  8$:   BIC     #BIT02,XCSR         ;CLEAR LOOP BACK MODE BIT
20160
20161 120134                            SLEND:                      ;LAST SLU TEST
```

```
                                        .SBTTL  TEST  - LED'S ON
                                        ;LED'S ON
                                        ;THIS TEST WILL INITIALIZE BDR TO CONTAIN A ROTATING PATTERN
                                        ;DISPLAYED IN LED'S.
                                        ;
                                        ;ROUTINE TEST
                                        ;.      WHILE A KEY NOT RECEIVED FROM KEYBOARD DO
                                        ;.      .       STALL ALLOWING TIME TO SEE PATTERN
                                        ;.      .       ROTATE LEFT TO LIGHT UP NEXT LED'S
                                        ;.      ENDDO
                                        ;ENDROUTINE

                                        ;;*****************************************************************
        120134  000004                  TST54:  SCOPE
        120136  052737  001000  177520          BIS     #1000,BCSR                      ;ENABLE HOB FOR APT
        120144  005005                          CLR     R5                              ;FLAG IN NO INTERRRUPT MODE
        120146  032737  000001  000052          BIT     #BIT00,@#52                     ;IF RUNNING IN CHAIN MODE
        120154  001427                          BEQ     1$                              ;SKIP PRINTOUTS
        120156  005737  001206                  TST     $PASS                           ;1ST PASS?
        120162  001024                          BNE     1$                              ;IF NOT, SKIP PRINTOUTS
        120164  122737  000001  001220          CMPB    #APTENV,$ENV                    ;APT MODE?
        120172  001420                          BEQ     1$                              ;YES, SKIP PRINTOUT'S
        120174  005105                          COM     R5                              ;CLEAR FLAG IN INTERRUPT MODE
        120176  104401  001175                  TYPE    ,$CRLF
        120202  104401  120332                  TYPE    ,LEDS                           ;IDENTIFY THE TEST
        120206  012737  120304  000060          MOV     #5$,@#60                        ;RECEIVE SLU VECTOR
        120214  012737  000340  000062          MOV     #340,@#62                       ;AT PRIORITY 7
        120222  052737  000100  177560          BIS     #BIT06,RCSR                     ;ENABLE INTERRUPTS
        120230  106427  000140                  MTPS    #140                            ;LOWER PRIORITY
        120234  012704  000006          1$:     MOV     #6,R4                           ;FOR EACH LOOP
        120240  012701  000076                  MOV     #76,R1                          ;START WITH 1
        120244  110137  177524          2$:     MOVB    R1,BDR                          ;TURN OFF FIRST LED
        120250  012703  000004                  MOV     #4,R3                           ;STALL DELAY
        120254  012702  177777          3$:     MOV     #177777,R2                      ;STALL DELAY
        120260  077201                  4$:     SOB     R2,4$                           ;WAIT A WHILE
        120262  077304                          SOB     R3,3$                           ;WAIT A WHILE
        120264  000261                          SEC                                     ;SET CARRY
        120266  006101                          ROL     R1                              ;GET ANOTHER LED
        120270  077413                          SOB     R4,2$                           ;DO A FEW TIMES
        120272  005705                          TST     R5                              ;RUNNING IN INTERACTIVE MODE?
        120274  001407                          BEQ     6$                              ;IF NOT, EXIT
        120276  104401  001170                  TYPE    ,$BELL                          ;
        120302  000754                          BR      1$                              ;REPEAT PATTERN
        120304  005737  177562          5$:     TST     RBUF                            ;READ BUFFER
        120310  062706  000004                  ADD     #4,SP                           ;ADJUST STACK
        120314  112737  000377  177524  6$:     MOVB    #377,BDR                        ;NO MORE
        120322  042737  001000  177520          BIC     #1000,BCSR                      ;DISABLE HOB FOR APT
        120330  000452                          BR      TST55           ;;EXIT TEST

        120332     012     015     124  LEDS:   .ASCII  <12><15>/THIS IS A TEST FOR ON-BOARD LED'S/<12><15>
        120335     110     111     123
        120340     040     111     123
        120343     040     101     040
        120346     124     105     123
        120351     124     040     106
        120354     117     122     040
        120357     117     116     055
```

```
TEST  - LED'S ON

        120362  102     117     101
        120365  122     104     040
        120370  114     105     104
        120373  047     123     012
        120376  015
 20212  120377  124     131     120          .ASCIZ  /TYPE ANY CHARACTER ON A KEYBOARD TO CONTINUE/<12><15>
        120402  105     040     101
        120405  116     131     040
        120410  103     110     101
        120413  122     101     103
        120416  124     105     122
        120421  040     117     116
        120424  040     101     040
        120427  113     105     131
        120432  102     117     101
        120435  122     104     040
        120440  124     117     040
        120443  103     117     116
        120446  124     111     116
        120451  125     105     012
        120454  015     000
 20213                                .EVEN
 20214
```

```
20216                                   .SBTTL  TEST  - MEMORY MAPPING
20217                                   ;MEMORY MAPPING
20218                                   ;THIS TEST WILL AUTOSIZE MEMORY IN 2K BYTES. EVERY PAGE WILL BE
20219                                   ;CHECKED FOR WHAT TYPE IT IS: Q-BUS, UNIBUS OR PMI BUS MEMORY BY
20220                                   ;SEEING HOW IT CAN BE CACHED.
20221                                   ;ROUTINE TEST
20222                                   ;.  LET 4=ADDRESS OF NON-EXISTENT_MEMORY
20223                                   ;.  IF KMCR<05-00> NE #1 THEN (NOT ALL UNIBUS MEMORY)
20224                                   ;.      TURN ON MMU AND REMAP THE PROGRAM AREA
20225                                   ;.      REPEAT
20226                                   ;.      .  DO FOR KDPAR0 FROM #0 TO #177600
20227                                   ;.      .       DO FOR R1 FROM #0 TO #20000 BY #4000
20228                                   ;.      .       .       READ (R1)
20229                                   ;.      .       .       IF ABORT_NON-EXISTENT_MEMORY NE 1
20230                                   ;.      .       .       .       MEMORY EXISTS
20231                                   ;.      .       .       .       READ (R1)+
20232                                   ;.      .       .       .       READ (R1)
20233                                   ;.      .       .       .       IF HIT/MISS EQ 2_HITS THEN
20234                                   ;.      .       .       .       .       PMI MEMORY
20235                                   ;.      .       .       .       ELSE
20236                                   ;.      .       .       .       .       IF HIT/MISS EQ HIT THEN
20237                                   ;.      .       .       .       .       .       Q-BUS MEMORY
20238                                   ;.      .       .       .       .       ELSE
20239                                   ;.      .       .       .       .       .       ERROR
20240                                   ;.      .       .       .       .       ENDIF
20241                                   ;.      .       .       .       ENDIF
20242                                   ;.      .       .       ENDIF
20243                                   ;.      .       .       LET ABORT_NON-EXISTENT_MEMORY=#0
20244                                   ;.      .       ENDDO
20245                                   ;.      .  ENDDO
20246                                   ;.      UNTILL ABORT_NON-EXISTANT_MEMORY EQ #1
20247                                   ;.  ENDIF
20248                                   ;.
20249                                   ;ENDROUTINE
20250                                   ;
20251                                   ;ROUTINE NON-EXISTENT_MEMORY
20252                                   ;.      LET ABORT_NON-EXISTENT_MEMORY=#1
20253                                   ;.      RETURN
20254                                   ;ENDROUTINE
20255
20256                                   ;;*****************************************************************
    120456  000004                      TST55:  SCOPE
20257 120460  032737  000001  000052            BIT     #BIT00,@#52                     ;CHAIN MODE?
20258 120466  001561                            BEQ     TST56           ;;IF SO, EXIT TEST
20259 120470  005737  001206                    TST     $PASS                           ;FIRST PASS?
20260 120474  001156                            BNE     TST56           ;;IF APT AND NOT FIRST PASS, EXIT TEST
20261 120476  122737  000001  001220            CMPB    #APTENV,$ENV                    ;APT MODE?
20262 120504  001552                            BEQ     TST56           ;;YES, SKIP PRINTOUT'S
20263                                   ;
20264                                   ; SETUP ALL REGISTERS FOR MAPPING
20265                                   ;
20266 120506  012737  000400  177746            MOV     #400,CCR                        ;FLUSH THE CACHE
20267 120514  013704  000004                    MOV     ERRVEC,R4                       ;STORE TIMEOUT VECTOR
20268 120520  012737  120670  000004            MOV     #7$,@#ERRVEC                    ;POINT TO PROGRAM AREA
20269 120526  012737  000340  000006            MOV     #340,ERRVEC+2                   ;AT PRIORITY 4
20270 120534  004737  134110                    JSR     PC,INITMM                       ;REMAP PROGRAM AREA
20271 120540  005037  172354                    CLR     KIPAR6                          ;USED FOR MAPPING MEMORY
```

TEST - MEMORY MAPPING

```
20272 120544  005002                          CLR     R2                      ;CLEAR PAGE COUNT FOR PMI
20273 120546  005003                          CLR     R3                      ;CLEAR PAGE COUNT FOR QBUS
20274 120550  005237  177572                  INC     MMR0                    ;ENABLE MEMORY MANGEMENT
20275 120554  052737  000020  172516          BIS     #BIT04,MMR3             ;ENABLE 22 BITS
20276 120562  000403                          BR      2$                      ;GO ACCESS
20277                                 ;
20278                                 ; TRY TO MAP ALL PAGES
20279                                 ;
20280 120564  062737  000200  172354  1$:     ADD     #200,KIPAR6             ;INCREMENT BY 4K WORDS
20281 120572  012701  140000          2$:     MOV     #140000,R1              ;ACCESS THRU KIPAR6
20282 120576  000402                          BR      4$                      ;START DOING IT
20283 120600  062701  003776          3$:     ADD     #3776,R1                ;INCREMENT BY 1K WORDS
20284 120604  005721                  4$:     TST     (R1)+                   ;ACCESS 1ST LOCATION
20285 120606  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20286 120614  005711                          TST     (R1)                    ;ACCESS 2ND LOCATION
20287 120616  013737  177752  111464          MOV     HITMIS,RECDAT           ;STORE REGISTER
20288 120624  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20289 120632  032737  000004  111464          BIT     #BIT02,RECDAT           ;LAST (R1) HIT?
20290 120640  001402                          BEQ     5$                      ;IF NOT, QBUS MEMORY
20291 120642  005202                          INC     R2                      ;INCREMENT 1K COUNT FOR PMI
20292 120644  000401                          BR      6$                      ;GO CONITNUE
20293 120646  005203                  5$:     INC     R3                      ;INCREMENT COUNT FOR QBUS MEM.
20294 120650  022701  154000          6$:     CMP     #154000,R1              ;LAST IN 4K PAGE?
20295 120654  101351                          BHI     3$                      ;IF NOT, BRANCH
20296 120656  022737  177600  172354          CMP     #177600,KIPAR6          ;2M BOUNDARY?
20297 120664  001337                          BNE     1$                      ;IF NOT, BRANCH
20298 120666  000402                          BR      8$                      ;IF 2M, DON'T TOUCH STACK
20299                                 ;
20300                                 ; MAPPING IS DONE, FIND OUT HOW MANY PAGES WERE THERE
20301                                 ;
20302 120670  005726                  7$:     TST     (SP)+                   ;ADJUST STACK
20303 120672  005726                          TST     (SP)+
20304 120674  010437  000004          8$:     MOV     R4,ERRVEC               ;RESTORE ERROR VECTOR
20305 120700  005037  177572                  CLR     MMR0                    ;DISABLE MEMORY MANGEMENT
20306 120704  042737  000020  172516          BIC     #BIT04,MMR3             ;DISABLE 22 BITS
20307 120712  005702                          TST     R2                      ;ANY PMI MEMORY?
20308 120714  001405                          BEQ     9$                      ;IF NOT, GO CHECK QBUS MEMORY
20309 120716  006302                          ASL     R2                      ;TRANSLATE TO BYTES
20310 120720  010246                          MOV     R2,-(SP)                ;STORE 1K # ON STACK
20311 120722  104405                          TYPDS                           ;TYPE # OF PAGES
20312 120724  104401  120750                  TYPE    ,MEMK                   ;TYPE ASCII
20313 120730  005703                  9$:     TST     R3                      ;ANY Q-BUS MEMORY?
20314 120732  001405                          BEQ     10$                     ;IF NOT, BRANCH
20315 120734  006303                          ASL     R3                      ;TRANSLATE TO BYTES
20316 120736  010346                          MOV     R3,-(SP)                ;STORE 1K # ON STACK
20317 120740  104405                          TYPDS                           ;TYPE # OF PAGES
20318 120742  104401  121000                  TYPE    ,MEMQ                   ;TYPE ASCII
20319 120746                          10$:
      120746  000431                          BR      TST56           ;;EXIT TEST
20320
20321 120750      113     040     102  MEMK:   .ASCIZ  /K BYTES OF PMI MEMORY/<12><15>
      120753      131     124     105
      120756      123     040     117
      120761      106     040     120
      120764      115     111     040
      120767      115     105     115
      120772      117     122     131
```

```
TEST  - MEMORY MAPPING

        120775  012     015     000
  20322 121000  113     040     102     MEMQ:   .ASCIZ  /K BYTES OF Q-BUS MEMORY/<12><15>
        121003  131     124     105
        121006  123     040     117
        121011  106     040     121
        121014  055     102     125
        121017  123     040     115
        121022  105     115     117
        121025  122     131     012
        121030  015     000
  20323                                 .EVEN
  20324
  20325                                 .SBTTL WRONG PARITY ABORT TEST
  20326                                 ;WRONG PARITY ABORT
  20327                                 ;THIS TEST VERIFIES A3ORT TO 114 USING PARITY OR ECC MEMORY CSR.
  20328                                 ;IF MORE THEN 1 CSR PRESENT, ALL OF THEM WILL BE WRITTEN AT THE
  20329                                 ;SAME TIME.
  20330                                 ;
  20331                                 ;ROUTINE TEST
  20332                                 ;.      SIZE FOR ALL POSSIBLE MEMORY CSR'S
  20333                                 ;.      STORE UP TO 16 CSR STARTING FROM TEMP
  20334                                 ;.      WRITE ALL WITH WRONG PARITY
  20335                                 ;.      WRITE TO 0
  20336                                 ;.      READ IT BACK
  20337                                 ;.      IF NO ABORT TO 114 THEN
  20338                                 ;.              ERROR IN PARITY ABORT LOGIC
  20339                                 ;.      ENDIF
  20340                                 ;.      RESTORE CSR'S
  20341                                 ;ENDROUTINE
  20342
  20343                                 ;;*****************************************************************
        121032  000004                  TST56:  SCOPE
  20344                                 ;
  20345                                 ; FIND OUT ALL POSSIBLE MEMORY CSR LOCATIONS UP TO 16
  20346                                 ;
  20347 121034  013701  000004          MOV     ERRVEC,R1               ;STORE TIMEOUT VECTOR
  20348 121040  012737  121076  000004  MOV     #2$,ERRVEC              ;POINT NEW TO PROGRAM
  20349 121046  012737  000340  000006  MOV     #340,ERRVEC+2           ;AT PRIORITY 7
  20350 121054  005004                  CLR     R4                      ;COUNT FOR CSR'S
  20351 121056  012702  172100          MOV     #172100,R2              ;FIRST POSSIBLE CSR
  20352 121062  012703  002740          MOV     #TEMP,R3                ;STORAGE LOCATION
  20353 121066  005712                  1$:     TST     (R2)            ;IS CSR THERE?
  20354 121070  010223                  MOV     R2,(R3)+                ;IF THERE, STORE
  20355 121072  005204                  INC     R4                      ;INCREMENT COUNT FOR CSR'S
  20356 121074  000402                  BR      3$                      ;BRANCH AROUND
  20357 121076  005726                  2$:     TST     (SP)+           ;RESTORE STACK
  20358 121100  005726                  TST     (SP)+
  20359 121102  062702  000002          3$:     ADD     #2,R2           ;POINT TO NEW CSR
  20360 121106  022702  172136          CMP     #172136,R2              ;ALL DONE?
  20361 121112  101365                  BHI     1$                      ;IF NOT, BRANCH
  20362 121114  010137  000004          MOV     R1,ERRVEC               ;RESTORE ERROR VECTOR
  20363 121120  052737  001000  177746  BIS     #BIT09,CCR              ;SET CACHE BYPASS
  20364                                 ;
  20365                                 ; WRITE ALL CSR'S WITH WRONG ECC CODE
  20366                                 ; NOTE: IN PARITY MEMORY THOSE BITS ARE READ ONLY AND
  20367                                 ;       DIAGNOSTIC MODE BIT FOR ECC IS THE SAME AS WRONG PARITY
  20368                                 ;
```

WRONG PARITY ABORT TEST

```
20369 121126  013701  000114                    MOV     @#114,R1                ;STORE PARITY ABORT VECTOR
20370 121132  012737  121220  000114            MOV     #6$,@#114               ;POINT NEW TO PROGRAM
20371 121140  012737  000340  000116            MOV     #340,@#116              ;AT PRIORITY 7
20372 121146  010402                            MOV     R4,R2                   ;STORE CSR COUNT
20373 121150  012703  002740                    MOV     #TEMP,R3                ;START WITH 1ST CSR
20374 121154  052773  000005  000000  4$:       BIS     #BIT02!BIT00,@0(R3)     ;DIAGNOSTIC OR WRONG PARITY
20375 121162  042733  003740                    BIC     #3740,@(R3)+            ;CLEAR <10-5> CHECK BITS
20376 121166  077406                            SOB     R4,4$                   ;DO FOR ALL CSR'S
20377 121170  005037  000000                    CLR     @#0                     ;CLEAR TEST LOCATION WITH WRONG PR
20378 121174  010204                            MOV     R2,R4                   ;RESTORE COUNTER
20379 121176  012703  002740                    MOV     #TEMP,R3                ;START WITH 1ST CSR
20380 121202  042733  000004          5$:       BIC     #BIT02,@(R3)+           ;CLEAR ALL WRONG PARITY
20381 121206  077203                            SOB     R2,5$                   ;IN ALL CSR'S
20382 121210  005737  000000                    TST     @#0                     ;READ BACK WRONG PARITY
20383 121214  104034                            ERROR   .34                     ;NO WRONG PARITY ABORT
20384 121216  000402                            BR      7$
20385 121220  005726                  6$:       TST     (SP)+                   ;ADJUST STACK
20386 121222  005726                            TST     (SP)+
20387 121224  012703  002740          7$:       MOV     #TEMP,R3                ;START WITH 1ST CSR
20388 121230  042733  000001          8$:       BIC     #BIT00,@(R3)+           ;CLEAR ALL WRONG PARITY
20389 121234  077403                            SOB     R4,8$                   ;IN ALL CSR'S
20390 121236  032737  100000  177744            BIT     #BIT15,MSER             ;ABORT REFLECTED IN MSER?
20391 121244  001004                            BNE     9$                      ;IF YES, BRANCH
20392 121246  012737  100000  001124            MOV     #100000,$GDDAT          ;
20393 121254  104035                            ERROR   .35                     ;MSER<15> NOT SET ON PARITY ABORT
20394 121256  042737  001000  177746  9$:       BIC     #BIT09,CCR              ;CLEAR CACHE BYPASS
20395 121264  005037  000000                    CLR     @#0                     ;CLEAR WRONG PARITY
20396 121270  005037  177744                    CLR     MSER                    ;CLEAR MSER
20397 121274  010137  000114                    MOV     R1,@#114                ;RESTORE PARITY ABORT
20398
20399
```

TEST - DMA TAG PARITY IN STANDALONE MODE

```
20401                                   .SBTTL  TEST  - DMA TAG PARITY IN STANDALONE MODE
20402                                   ;CHECK DMA TAG STORE PARITY BIT.
20403                                   ;ROUTINE TEST
20404                                   ;.      CACHE DMA_PARITY
20405                                   ;.      LET BCSR<08>=#1
20406                                   ;.      REPORT ALL ERRORS
20407                                   ;ENDROUTINE
20408                                   ;
20409                                   ;ROUTINE DMA_PARITY
20410                                   ;.      GENERATE PARITY ERRORS
20411                                   ;.      CHECK MSER<13>
20412                                   ;.      LET BCSR<08>=#0
20413                                   ;ENDROUTINE
20414
20415                                   ;;*****************************************************************
     121300  000004                     TST57:  SCOPE
20416 121302  000240                             NOP
20417 121304  005737  003032                     TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
20418 121310  001002                             BNE     99$                     ; NOT YET
20419 121312  000240                             NOP                             ; DEBUG AID
20420 121314  000456                             BR      TST60           ;;GO TO NEXT TEST
20421 121316  000240                     99$:    NOP
20422                                   ;
20423                                   ; ALLOCATE CODE IN CACHE
20424                                   ;
20425 121320  012702  121354                     MOV     #DMAPAR,R2                      ;POINT TO STANDALONE CODE
20426 121324  042737  001000  177520             BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
20427 121332  005722                     1$:     TST     (R2)+                           ;ALLOCATE IN CACHE
20428 121334  022702  121414                     CMP     #DPAREN,R2                      ;LAST ADDRESS?
20429 121340  001374                             BNE     1$                              ;IF NOT, BRANCH
20430 121342  005737  002740                     TST     TEMP                            ;ALLOCATE TEST LOCATION
20431 121346  013737  177520  002730             MOV     BCSR,SAVBR                      ;SAVE BCSR
20432                                   ;
20433                                   ; IN STANDALONE MODE TRY TO VERIFY RESPONSE TO PARITY ERRORS
20434                                   ;
20435 121354  052737  000400  177520     DMAPAR: BIS     #BIT08,BCSR                     ;SET STANDALONE MODE BIT
20436 121362  052737  002001  177746             BIS     #BIT10!BIT00,CCR                ;WRITE WRONG TAG PARITY
20437 121370  005037  002740                     CLR     TEMP                            ;WRITE HIT WITH WRONG PARITY
20438 121374  013705  177744                     MOV     MSER,R5                         ;STORE MSER
20439 121400  042737  002000  177746             BIC     #BIT10,CCR                      ;CLEAR WRONG PARITY BIT
20440 121406  042737  000400  177520     2$:     BIC     #BIT08,BCSR                     ;CLEAR STANDALONE BIT
20441                                   ;
20442                                   ; RETURN FROM STANDALONE MODE
20443                                   ;
20444 121414                            DPAREN:
20445 121414  022705  060020                     CMP     #60020,R5                       ;WRONG DMA PARITY?
20446 121420  001401                             BEQ     4$                              ;IF OK, BRANCH
20447 121422  104117                             ERROR   +117                            ;MSER NOT SET IN STANDALONE MODE
20448 121424  005037  177744             4$:     CLR     MSER
20449 121430  012737  000400  177746             MOV     #400,CCR                        ;FLUSH THE CACHE
20450 121436  013737  002730  177520             MOV     SAVBR,BCSR                      ;RESTORE BCSR
20451 121444  052737  001000  177520             BIS     #1000,BCSR                      ;ENABLE HALT ON BREAK
20452
```

TEST - DMA TAG PARITY W/O STANDALONE MODE

```
                                .SBTTL  TEST  - DMA TAG PARITY W/O STANDALONE MODE
                                ;CHECK DMA TAG PARITY BIT WITHOUT GOING INTO STANDALONE MODE.
                                ;
                                ;CCR    <10>    WRITE WRONG TAG PARITY
                                ;
                                ;ROUTINE TEST
                                ;.      LET CCR<10>=#1
                                ;.      ALLOCATE LOCATION IN CACHE
                                ;.      INITIATE DMA WRITE TRANSFERS
                                ;.      IF MSER<04> NE #1 THEN
                                ;.          ERROR
                                ;.      ENDIF
                                ;ENDROUTINE
                                ;;*****************************************************************
    121452  000004              TST60:  SCOPE
20469 121454  000240                    NOP
20470 121456  005737  003032            TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
20471 121462  001002                    BNE     99$                     ; NOT YET
20472 121464  000240                    NOP                             ; DEBUG AID
20473 121466  000471                    BR      TST61           ;;GO TO NEXT TEST
20474 121470  000240              99$:  NOP
20475
20476 121472  032737  000200  000052    BIT     #BIT07,@#52                     ;UFD MODE?
20477 121500  001402                    BEQ     110$                            ;IF NOT, BRANCH
20478 121502  000137  135446            JMP     $EOP                            ;OTHERWISE, NEXT PASS
20479 121506  005737  002664      110$: TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
20480 121512  001457                    BEQ     TST61           ;;IF NOT, EXIT TEST
20481                                   ;
20482                                   ; ALLOCATE TEST IN CACHE
20483                                   ;
20484 121514  012703  121514      11$:  MOV     #.,R3                           ;START WITH CURRENT INSTRUCTION
20485 121520  005723              10$:  TST     (R3)+                           ;READ A WORD
20486 121522  022703  121632            CMP     #4$,R3                          ;ALL DONE?
20487 121526  001374                    BNE     10$                             ;IF NOT, CONTINUE
20488                                   ;
20489                                   ; WRITE A WORD WITH WRONG PARITY
20490                                   ;
20491 121530  013737  000114  001160  1$: MOV     @#114,$TMP0                     ;STORE PARITY VECTOR
20492 121536  012737  121604  000114      MOV     #2$,@#114                       ;POINT TO TEST AREA
20493 121544  052737  002000  177746      BIS     #BIT10,CCR                      ;WRITE WRONG TAG PARITY
20494 121552  005037  002740              CLR     TEMP                            ;WRITE MISS WITH WRONG TAG PARITY
20495 121556  042737  002000  177746      BIC     #BIT10,CCR                      ;CLEAR WRONG TAG PARITY
20496 121564  052737  000200  177746      BIS     #BIT07,CCR                      ;PARITY ABORT
20497 121572  005000                    CLR     R0                              ;FLAG TO DO 1 TRANSFER
20498 121574  004737  134704            JSR     PC,DMATRN                       ;DO DMA WRITE TO TEMP THRU Q22BE
20499 121600  104116                    ERROR   +116                            ;NO PARITY ABORT
20500 121602  000413                    BR      4$                              ;BRANCH TO TEST MSER
20501 121604  013704  177744      2$:   MOV     MSER,R4                         ;STORE REGISTER
20502 121610  005037  177744            CLR     MSER                            ;CLEAR MSER
20503 121614  005726                    TST     (SP)+                           ;
20504 121616  005726                    TST     (SP)+                           ;RESTORE STACK
20505 121620  005726                    TST     (SP)+
20506 121622  032704  000020      3$:   BIT     #BIT04,R4                       ;MSER OK?
20507 121626  001001                    BNE     4$                              ;IF SET, BRANCH
20508 121630  104116                    ERROR   +116                            ;MSER<4> NOT SET
20509 121632  005037  177744      4$:   CLR     MSER                            ;CLEAR MSER
```

TEST - DMA TAG PARITY W/O STANDALONE MODE

```
 20510 121636  012737  000400  177746          MOV     #400,CCR                ;FLUSH CACHE
 20511 121644  013737  001160  000114          MOV     $TMP0,@#114             ;RESTORE VECTOR
```

TEST - DMA WRITE HIT CYCLES

```
                                .SBTTL  TEST  - DMA WRITE HIT CYCLES
                                ;CHECK THAT DMA WRITE HITS INVALIDATE CACHE.
                                ;ROUTINE TEST
                                ;.      ALLOCATE A LOCATION IN CACHE
                                ;.      INITIATE DMA WRITE TO THIS LOCATION
                                ;.      READ THIS LOCATION BACK
                                ;.      IF IT IS CHANGED OR HIT/MISS EQ HIT
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      WRITE TO THE FIRST 16 LOCATION THEIR ADDRESS
                                ;.      DO BLOCK MODE TRANSFER TO THOSE LOCATIONS
                                ;.      START READ WITH THE LAST LOCATION
                                ;.      IF RECORD ANY HITS THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      INITIATE READ DMA TO THE SAME TEST LOCATIONS
                                ;.      READ THEM BACK
                                ;.      IF HIT/MISS NE HIT THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;ENDROUTINE

                                ;;**********************************************************************
    121652  000004              TST61:  SCOPE
20536 121654  000240                    NOP
20537 121656  005737  003032            TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
20538 121662  001003                    BNE     99$                     ; NOT YET
20539 121664  000240                    NOP                             ; DEBUG AID
20540 121666  000137  122620            JMP     ALLEND                  ; YES SKIP THIS
20541 121672  000240              99$:  NOP
20542
20543 121674  032737  000200  000052    BIT     #BIT07,@#52             ;UFD MODE?
20544 121702  001402                    BEQ     1$                      ;IF NOT, BRANCH
20545 121704  000137  135446            JMP     $EOP                    ;OTHERWISE, NEXT PASS
20546 121710  005737  002664      1$:   TST     CSR1                    ;AT LEAST 1 Q22BE?
20547 121714  001002                    BNE     2$                      ;IF YES, BRANCH
20548 121716  000137  135446            JMP     $EOP                    ;OTHERWISE, NEXT PASS
20549                           ;
20550                           ; TRY TO DO DMA WRITE AT ALL DIFFERENT PRIORITIES
20551                           ;
20552 121722  012702  000340      2$:   MOV     #340,R2                 ;START WITH 7
20553 121726  000402                    BR      4$                      ;GO DO IT
20554 121730  162702  000040      3$:   SUB     #40,R2                  ;LOWER PRIORITY
20555 121734  005037  002740      4$:   CLR     TEMP                    ;CLEAR TEST LOCATION
20556 121740  005000                    CLR     R0                      ;DO JUST 1 WORD
20557 121742  106402                    MTPS    R2                      ;LOWER PRIORITY
20558 121744  004737  134704            JSR     PC,DMATRN               ;DO DATO
20559 121750  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20560 121756  005737  002740            TST     TEMP                    ;STILL CACHED?
20561 121762  013737  177752  111464    MOV     HITMIS,RECDAT           ;STORE HIT/MISS
20562 121770  052737  001000  177520    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20563 121776  032737  000004  111464    BIT     #BIT02,RECDAT           ;LAST ACCESS HIT?
20564 122004  001401                    BEQ     5$                      ;IF MISS, BRANCH
20565 122006  104120                    ERROR   +120
20566 122010  022737  012525  002740  5$:   CMP     #12525,TEMP             ;DATO OK?
20567 122016  001401                    BEQ     6$                      ;IF SO, BRANCH
20568 122020  104123                    ERROR   +123                    ;DATO
```

TEST - DMA WRITE HIT CYCLES

```
20569 122022  005702                    6$:     TST     R2                      ;LAST PRIORITY 0?
20570 122024  001341                            BNE     3$                      ;IF NOT, BRANCH
20571 122026  106427  000340                    MTPS    #340                    ;RESTORE PRIORITY
20572                                   ;
20573                                   ;
20574                                   ; VERIFY THAT BYPASS WITH DMA DATI INVALIDATES CACHE
20575                                   ;
20576 122032  012737  001000  177746    8$:     MOV     #BIT09,CCR              ;SET BYPASS
20577 122040  004737  134704            JSR     PC,DMATRN               ;DO DMA DATI
20578 122044  005037  177746            CLR     CCR                     ;CLEAR BYPASS
20579 122050  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20580 122056  005737  002740            TST     TEMP                    ;IN CACHE?
20581 122062  013737  177752  111464    MOV     HITMIS,RECDAT           ;STORE REGISTER
20582 122070  052737  001000  177520    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20583 122076  032737  000004  111464    BIT     #BIT02,RECDAT           ;TEMP WAS A HIT?
20584 122104  001401                    BEQ     DATBO                   ;IF NOT, BRANCH
20585 122106  104123                    ERROR   +123
20586                                   ;
20587                                   ; DO DATBO
20588                                   ;
20589 122110  012701  000010    DATBO:  MOV     #10,R1                  ;COUNTER FOR 8
20590 122114  012702  002740            MOV     #TEMP,R2                ;START WITH TEMP
20591 122120  005022                    1$:     CLR     (R2)+                   ;CLEAR ALL 16
20592 122122  077102                    SOB     R1,1$                   ;DO ALL 16
20593 122124  005200                    INC     R0                      ;FLAG BLOCK MODE
20594 122126  012777  002740  060534    MOV     #TEMP,@BA               ;LOAD DMA ADDRESS
20595 122134  012777  177770  060530    MOV     #177770,@WC             ;DO 8 WORDS
20596 122142  012777  001701  060514    MOV     #1701,@CSR1             ;16 WORDS FROM 32K
20597 122150  004737  134704            JSR     PC,DMATRN               ;DO DATBO
20598 122154  032777  010000  060504    BIT     #BIT12,@CSR2            ;NO BLOCK MODE SLAVE?
20599 122162  001401                    BEQ     2$                      ;IF NOT, BRANCH
20600 122164  104123                    ERROR   +123                    ;NO BLOCK MODE SLAVE
20601 122166  012701  000004            2$:     MOV     #4,R1                   ;COUNTER FOR 4 LOCATIONS
20602 122172  012702  002740            MOV     #TEMP,R2                ;START WITH TEMP
20603 122176  042737  001000  177520    3$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20604 122204  005712                    TST     (R2)                    ;ACCESS A LOCATION
20605 122206  013737  177752  111464    MOV     HITMIS,RECDAT           ;STORE REGISTER
20606 122214  052737  001000  177520    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20607 122222  032737  000004  111464    BIT     #BIT02,RECDAT           ;HIT?
20608 122230  001401                    BEQ     4$                      ;IF NOT, BRANCH
20609 122232  104121                    ERROR   +121                    ;DMA DOES NOT INVALIDATE
20610 122234  022722  012525            4$:     CMP     #12525,(R2)+            ;DATO OK?
20611 122240  001401                    BEQ     5$                      ;IF SO,BRANCH
20612 122242  104123                    ERROR   +123                    ;IN DMA
20613 122244  062702  000002            5$:     ADD     #2,R2                   ;DO IN 2 WORDS
20614 122250  077126                    SOB     R1,3$                   ;DO ALL 16
20615
20616                                   ;
20617                                   ; DO 4K OF DATO AND CHECK FOR MISS
20618                                   ;
20619 122252  004737  134110            JSR     PC,INITMM               ;SET UP MMU
20620 122256  012737  002000  172354    MOV     #2000,KIPAR6            ;START AT 32K
20621 122264  042737  100000  172314    BIC     #BIT15,KIPDR6           ;NO BYPASS
20622 122272  052737  000001  177572    BIS     #BIT00,MMR0             ;ENABLE MMU
20623 122300  012737  122440  000004    MOV     #DATI,@#4               ;IF 32K NXW
20624 122306  012702  140000            MOV     #140000,R2              ;START WITH 32K
20625 122312  005712                    TST     (R2)                    ;EXIT
```

```
20626 122314  012701  010000              MOV     #10000,R1               ;COUNTER FOR 4K
20627 122320  005022              6$:     CLR     (R2)+                   ;CLEAR ALL 16
20628 122322  077102                      SOB     R1,6$                   ;DO ALL 16
20629 122324  005200                      INC     R0                      ;FLAG BLOCK MODE
20630 122326  012777  000000  060334      MOV     #0,@BA                  ;LOAD DMA ADDRESS
20631 122334  012777  170000  060330      MOV     #-10000,@WC             ;DO 4K
20632 122342  012777  003701  060314      MOV     #3701,@CSR1             ;16 WORDS FROM 32K
20633 122350  004737  134704              JSR     PC,DMATRN               ;DO DATBO
20634 122354  012701  004000        7$:   MOV     #4000,R1                ;COUNTER FOR 4K LOCATIONS
20635 122360  012702  140000              MOV     #140000,R2              ;START WITH 0
20636 122364  042737  001000  177520  8$:   BIC     #1000,@CSR            ;DISABLE HALT ON BREAK
20637 122372  005712                      TST     (R2)                    ;ACCESS A LOCATION
20638 122374  013737  177752  111464      MOV     HITMIS,RECDAT           ;STORE REGISTER
20639 122402  052737  001000  177520      BIS     #1000,@CSR              ;ENABLE HALT ON BREAK
20640 122410  032737  000004  111464      BIT     #BIT02,RECDAT           ;HIT?
20641 122416  001401                      BEQ     9$                      ;IF NOT, BRANCH
20642 122420  104121                      ERROR   +121                    ;DMA DOES NOT INVALIDATE
20643 122422  022722  012525        9$:   CMP     #12525,(R2)+            ;DATO OK?
20644 122426  001401                      BEQ     10$                     ;IF SO,BRANCH
20645 122430  104123                      ERROR   +123                    ;IN DMA
20646 122432  062702  000002        10$:  ADD     #2,R2                   ;DO IN 2 WORDS
20647 122436  077126                      SOB     R1,8$                   ;DO ALL OF THEM
20648                               ;
20649                               ; CHECK DATI
20650                               ;
20651 122440  005037  177572        DATI: CLR     MMR0                    ;DISABLE MMU
20652 122444  012706  001100              MOV     #1100,SP                ;RESTORE STACK
20653 122450  012737  135056  000004      MOV     #TOUT,@#4               ;AND TIMEOUT
20654 122456  012737  052525  002740      MOV     #52525,TEMP             ;LOAD MEMORY
20655 122464  005000                      CLR     R0                      ;JUST 1 WORD
20656 122466  004737  134766              JSR     PC,DMARD                ;DO DATI
20657 122472  022777  052525  060174      CMP     #52525,@DATA            ;DATI OK?
20658 122500  001401                      BEQ     11$                     ;IF YES, BRANCH
20659 122502  104123                      ERROR   +123                    ;DATI
20660                               ;
20661                               ; DO DATBI
20662                               ;
20663 122504  012701  000010        11$:  MOV     #10,R1                  ;COUNTER FOR 16 LOCATIONS
20664 122510  012702  002740              MOV     #TEMP,R2                ;START WITH TEMP
20665 122514  012703  002740              MOV     #TEMP,R3                ;START WITH TEMP
20666 122520  010322                12$:  MOV     R3,(R2)+                ;PUT ADDRESSES
20667 122522  005723                      TST     (R3)+                   ;INCREMENT ADDRESS
20668 122524  077103                      SOB     R1,12$                  ;DO ALL 16
20669 122526  005200                      INC     R0                      ;FLAG BLOCK MODE
20670 122530  004737  134766              JSR     PC,DMARD                ;DO DATBI
20671 122534  032777  010000  060124      BIT     #BIT12,@CSR2            ;NO BLOCK MODE SLAVE?
20672 122542  001401                      BEQ     13$                     ;IF NO, BRANCH
20673 122544  104123                      ERROR   +123
20674 122546  022777  002756  060120  13$:  CMP   #TEMP+16,@DATA          ;DATI OK?
20675 122554  001401                      BEQ     14$                     ;IF SO, BRANCH
20676 122556  104123                      ERROR   +123                    ;IN DATIB
20677 122560  042737  001000  177520  14$:  BIC   #1000,@CSR             ;DISABLE HALT ON BREAK
20678 122566  005737  002740              TST     TEMP                    ;ACCESS
20679 122572  013737  177752  111464      MOV     HITMIS,RECDAT           ;STORE REGISTER
20680 122600  052737  001000  177520      BIS     #1000,@CSR              ;ENABLE HALT ON BREAK
20681 122606  032737  000004  111464      BIT     #BIT02,RECDAT           ;HIT?
20682 122614  001001                      BNE     15$                     ;IF YES, BRANCH
```

```
20683 122616  104123                          ERROR   +123
20684 122620                          15$:
20685 122620                          ALLEND:
```

```
20687                                   .SBTTL  TEST  - DIFFERENT LEVELS OF INTERRUPTS
20688                                   ;DIFFERENT LEVELS OF INTERRUPTS
20689                                   ;THIS TEST WILL PROGRAM Q22 BUS EXERCISER TO INTERRUPT AT DIFFERENT
20690                                   ;LEVELS.  ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS AND
20691                                   ;PIRQ'S WILL BE TESTED.
20692                                   ;
20693                                   ;CHECK DIFFERENT LEVELS OF INTERRUPTS.
20694                                   ;ROUTINE TEST
20695                                   ;.      SET VECTOR TO INTERRUPT_DMA
20696                                   ;.      FOR INTERRUPTS FROM 4 TO 7 DO
20697                                   ;.      .       ENABLE INTERRUPTS
20698                                   ;.      .       SET PRIORITY=INTERUPT
20699                                   ;.      .       IF INTERRUPT_FLAG SET THEN
20700                                   ;.      .       .       ERROR
20701                                   ;.      .       ENDIF
20702                                   ;.      .       ENABLE INTERRUPTS
20703                                   ;.      .       SET PRIORITY=INTERRUPT-1
20704                                   ;.      .       IF INTERRUPT_FLAG NOTSET THEN
20705                                   ;.      .       .       ERROR
20706                                   ;.      .       ENDIF
20707                                   ;.      .       LET INTERRUPT_DMA=0
20708                                   ;.      ENDDO
20709                                   ;ENDROUTINE
20710                                   ;
20711                                   ;ROUTINE INTERUPT_DMA
20712                                   ;.      LET INTERRUPT_FLAG=1
20713                                   ;RETURN
20714                                   ;ENDROUTINE
20715
20716                                   ;;*****************************************************************
        122620  000004                  TST62:  SCOPE
20717   122622  032737  000200  000052          BIT     @#BIT07,@#52                    ;UFD MODE?
20718   122630  001122                  BNE     TST63           ;;IF SO, EXIT TEST
20719   122632  005737  002664          TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
20720   122636  001517                  BEQ     TST63           ;;IF NOT, EXIT TEST
20721                                   ;
20722                                   ; SETUP INITIAL PRIORITY TO 7
20723                                   ;
20724   122640  013703  002664          MOV     CSR1,R3                         ;DO FOR FIRST FOUND Q22BE
20725   122644  012777  122740  060024  MOV     #5$,@VQBE1                      ;POINT INTERRUPT VECTOR TO PROGRAM
20726   122652  012777  000340  060020  MOV     #340,@VQPR1                     ;AT PRIORITY 7
20727   122660  012700  003002          MOV     #Q22EN,R0                       ;START WITH 7 FOR INTERRUPTS
20728   122664  012701  000340          MOV     #340,R1                         ;LOW BOUNDARY FOR NO INTERRUPTS
20729   122670  000402                  BR      2$                              ;TRY TO DO IT FOR FIRST
20730   122672  162701  000040  1$:     SUB     #40,R1                          ;LOWER LOW BOUNDARY
20731                                   ;
20732                                   ; CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN BR
20733                                   ;
20734   122676  012737  000340  001160  2$:     MOV     #340,$TMP0                      ;TOP PRIORITY FOR NO INTERRUPTS
20735   122704  000403                  BR      4$                              ;DO FIRST ONE
20736   122706  162737  000040  001160  3$:     SUB     #40,$TMP0                       ;DO AT NEXT LEVEL
20737   122714  106437  001160  4$:     MTPS    $TMP0                           ;SET PRIORITY NOT TO INTERRUPT
20738   122720  004737  134664          JSR     PC,Q22INT                       ;ENABLE INTERRUPTS
20739   122724  012077  057736          MOV     (R0)+,@CSR2                     ;CLEAR GO BIT
20740   122730  000240                  NOP
20741   122732  000240                  NOP
20742   122734  000240                  NOP
```

TEST - DIFFERENT LEVELS OF INTERRUPTS

```
20743 122736  000403                        BR      6$                      ;IF NO INTERUPT, BRANCH
20744 122740  104126                5$:     ERROR   .126                    ;INTERRUPTS HAPPEN
20745 122742  005726                        TST     (SP)+                   ;RESTORE STACK
20746 122744  005726                        TST     (SP)+
20747 122746  020137  001160        6$:     CMP     R1,$TMP0                ;LAST ONE?
20748 122752  001355                        BNE     3$                      ;IF NOT BRANCH
20749 122754  022710  000002                CMP     #2,(R0)                 ;AT BR4?
20750 122760  001344                        BNE     1$                      ;IF NOT LAST ONE, BRANCH
20751                               ;
20752                               ; INTERRUPT AT ALL LEVELS
20753                               ;
20754 122762  012777  123056  057706 INQ22:  MOV     #5$,@VQBE1              ;POINT INTERRUPT VECTOR TO PROGRAM
20755 122770  012777  000340  057702        MOV     #340,@VQPR1             ;AT PRIORITY 7
20756 122776  012700  003002                MOV     #Q22EN,R0               ;START WITH 7 FOR INTERRUPTS
20757 123002  012701  000300                MOV     #300,R1                 ;TOP BOUNDARY FOR INTERRUPTS
20758 123006  000402                        BR      2$                      ;TRY TO DO IT FOR FIRST
20759 123010  162701  000040        1$:     SUB     #40,R1                  ;LOWER TOP BOUNDARY
20760                               ;
20761                               ; CHECK THAT INTERRUPTS HAPPEN AT PRIORITY LOWER THAN BR
20762                               ;
20763 123014  010137  001160        2$:     MOV     R1,$TMP0                ;PRIORITY FOR INTERRUPTS
20764 123020  000403                        BR      4$                      ;DO FIRST ONE
20765 123022  162737  000040  001160 3$:     SUB     #40,$TMP0               ;DO AT NEXT LEVEL
20766 123030  106437  001160        4$:     MTPS    $TMP0                   ;SET PRIORITY NOT TO INTERRUPT
20767 123034  004737  134664                JSR     PC,Q22INT               ;ENABLE INTERRUPTS
20768 123040  011077  057622                MOV     (R0),@CSR2              ;CLEAR GO BIT
20769 123044  000240                        NOP                             ;WAIT A WHILE
20770 123046  000240                        NOP
20771 123050  000240                        NOP
20772 123052  104126                        ERROR   .126                    ;INTERRUPTS DON'T HAPPEN
20773 123054  000402                        BR      6$                      ;DON'T RESTORE STACK
20774 123056  005726                5$:     TST     (SP)+                   ;RESTORE STACK
20775 123060  005726                        TST     (SP)+
20776 123062  005737  001160        6$:     TST     $TMP0                   ;LAST ONE 0?
20777 123066  001355                        BNE     3$                      ;IF NOT BRANCH
20778 123070  022720  000002                CMP     #2,(R0)+                ;AT BR4?
20779 123074  001345                        BNE     1$                      ;IF NOT LAST ONE, BRANCH
20780
20781
```

TEST - ARBITRATION BETWEEN PIRQ'S AND INTERRUPTS

```
                                .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

                                ;;*****************************************************************
 123076  000004                TST63:  SCOPE
123100  032737  000200  000052          BIT     #BIT07,@#52                     ;UFD MODE?
123106  001065                        BNE     TST64           ;;IF SO, EXIT TEST
123110  005737  002664                TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
123114  001462                        BEQ     TST64           ;;IF NOT, EXIT TEST
123116  012777  123224  057552        MOV     #3$,@VQBE1                      ;SETUP Q22BE VECTOR
123124  012777  000340  057546        MOV     #340,@VQPR1                     ;AT PRIORITY 7
123132  012737  123234  000240        MOV     #4$,PIRQVEC                     ;SETUP PIRQ VECTOR
123140  012737  000340  000242        MOV     #340,PIRQVEC+2                  ;AT PRIORITY 7
123146  012700  003002                MOV     #Q22EN,R0                       ;POINT THRU PRIORITIES FOR Q22BE
123152  012704  123252                MOV     #PIRQT,R4                       ;POINTER THRU PIRQ'S
123156  013703  002664                MOV     CSR1,R3                         ;DO FOR FIRST Q22BE
123162  012702  000300                MOV     #300,R2                         ;START WITH CPU PRIORITY AT 7
123166  000402                        BR      2$                              ;DO FIRST ONE
123170  162702  000040        1$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
123174  106427  000340        2$:     MTPS    #340                            ;RAISE PRIORITY TO 7
123200  012437  177772                MOV     (R4)+,PIRQ                      ;SET PRIORITY FOR PIRQ'S
123204  004737  134664                JSR     PC,Q22INT                       ;INITIALISE Q22BE TO INTERRUPT
123210  012077  057452                MOV     (R0)+,@CSR2                     ;SET DONE BIT
123214  106402                        MTPS    R2                              ;LOWER PRIORITY
123216  000240                        NOP
123220  000240                        NOP
123222  000240                        NOP
123224  104124        3$:             ERROR   +124                            ;PIRQ'S DON'T TAKE OVER BIRQ'S
123226  005726                        TST     (SP)+                           ;CLEAN UP STACK
123230  005726                        TST     (SP)+
123232  000402                        BR      5$                              ;BRANCH AROUND PIRQ INTERRUPT
123234  005726        4$:             TST     (SP)+                           ;CLEAN UP STACK
123236  005726                        TST     (SP)+
123240  022702  000140        5$:     CMP     #140,R2                         ;PRIORITY 3 LAST ONE?
123244  001351                        BNE     1$                              ;IF NOT BRANCH
123246  005037  177772                CLR     PIRQ                            ;CLEAR ANY REQUESTS

123252  100000  040000  020000  PIRQT: .WORD  100000,40000,20000,10000        ;PIRQ'S<7-4>
 123260  010000
```

TEST - POWER DOWN TEST

```
                                        .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

                                        ;;*****************************************************************
 123262  000004                       TST64:  SCOPE
 123264  032737  000200  000052                BIT     #BIT07,@#52                     ;UFD MODE?
 123272  001033                               BNE     TST65           ;;EXIT TEST IN UFD  MODE
 123274  005737  002664                       TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
 123300  001430                               BEQ     TST65           ;;IF NOT, EXIT TEST
 123302  013737  000024  001160              MOV     PWRVEC,$TMP0                    ;SAVE POWER UP VECTOR
 123310  012737  123344  000024              MOV     #1$,PWRVEC                      ;POINT NEW TO PROGRAM
 123316  012737  000340  000026              MOV     #340,PWRVEC+2                   ;AT PRIORITY 7
 123324  012777  000040  057334              MOV     #BIT05,@CSR2                    ;DO POWER DOWN
 123332  000240                               NOP
 123334  005077  057326                       CLR     @CSR2                           ;CLEAR POWER DOWN BIT
 123340  104125                               ERROR   +125                            ;NO POWER DOWN TRAP
 123342  000404                               BR      2$                              ;SKIP RESTORING STACK
 123344  005077  057316          1$:         CLR     @CSR2                           ;CLEAR POWER DOWN BIT
 123350  005726                               TST     (SP)+                           ;RESTORE STACK POINTER
 123352  005726                               TST     (SP)+
 123354  013737  001160  000024  2$:         MOV     $TMP0,PWRVEC                    ;RESTORE POWER VECTOR
```

TEST - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS

```
                                        .SBTTL TEST    - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS
                                        ;IF TWO Q22BE ARE AVAILABLE, THIS TEST WILL CHECK THAT HIGHER LEVEL
                                        ;INTERRUPT REQUESTS TAKE PRIORITY OVER LOWER LEVEL ONES
                                        ;ROUTINE TEST
                                        ;.      IF UFD OR 2ND Q22BE NOT AVAILABLE THEN
                                        ;.      .       EXIT TEST
                                        ;.      ENDIF
                                        ;.      DO FOR PRIORITY_LEVELS FROM 4 TO 6
                                        ;.      .       SET UP 1ST Q22BE TO INTERRUPT AT PRIORITY_LEVEL
                                        ;.      .       SET UP 2ND Q22BE TO INTERRUPT AT PRIORITY_LEVEL+1
                                        ;.      .       SET UP CPU PRIORITY TO PRIORITY_LEVEL-1
                                        ;.      .       IF INTERRUPTS FORM 1ST Q22BE HAPPENED BEFORE 1ST THEN
                                        ;.      .       .       ERROR
                                        ;.      .       ENDIF
                                        ;.      ENDDO
                                        ;ENDROUTINE

                                        ;;*****************************************************************
    123362  000004                      TST65:  SCOPE
20890 123364  032737  000200  000052            BIT     @#BIT07,@#52                    ;UFD MODE?
20891 123372  001064                    BNE     TST66           ;;IF SO, EXIT TEST
20892 123374  005737  002704            TST     CSR12                                   ;SECOND Q22BE AVAILABLE?
20893 123400  001002                    BNE     1$                                      ;IF YES, GO DO TEST
20894 123402  000137  135446            JMP     $EOP                                    ;OTHERWISE, GOTO EOP
20895                                   ;
20896                                   ; INITIALISE VECTORS FOR Q22BE'S
20897                                   ;
20898 123406  012777  123522  057262  1$:     MOV     #4$,@VQBE1                      ;VECTOR FOR 1ST ONE
20899 123414  012777  000340  057256          MOV     #340,@VQPR1                     ;AT PRIORITY 7
20900 123422  012777  123532  057266          MOV     #5$,@VQBE2                      ;VECTOR FOR 2ND ONE
20901 123430  012777  000340  057262          MOV     #340,@VQPR2                     ;AT PRIORITY 7
20902 123436  012700  003004                  MOV     #Q22EN+2,R0                     ;POINTER FOR Q22BE BR'S
20903 123442  012702  000240                  MOV     #240,R2                         ;CPU AT 5
20904 123446  000402                    BR      3$                                      ;START DOING ARBITRATION
20905                                   ;
20906                                   ; DO FOR CPU PRIORITIES 5-3
20907                                   ;
20908 123450  162702  000040          2$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
20909 123454  106427  000340          3$:     MTPS    #340                            ;CPU AT 7
20910 123460  013703  002704                  MOV     CSR12,R3                        ;Q22BE 2 AT HIGHER BR
20911 123464  004737  134664                  JSR     PC,Q22INT                       ;INITIALISE TO INTERRUPTS
20912 123470  011077  057172                  MOV     (R0),@CSR2                      ;START 1ST ONE
20913 123474  013703  002664                  MOV     CSR1,R3                         ;Q22BE 1 AT LOWER BR
20914 123500  005740                          TST     -(R0)                           ;HIGHER PRIORITY
20915 123502  004737  134664                  JSR     PC,Q22INT                       ;INITIALISE
20916 123506  012077  057174                  MOV     (R0)+,@CSR22                    ;START 2ND ONE
20917 123512  106402                          MTPS    R2                              ;LOWER CPU PRIORITY
20918 123514  000240                          NOP                                     ;WAIT A WHILE
20919 123516  000240                          NOP
20920 123520  000240                          NOP
20921 123522  104126                  4$:     ERROR   +126                            ;INTERRUPTS IN WRONG ORDER
20922 123524  005726                          TST     (SP)+                           ;RESTORE STACK
20923 123526  005726                          TST     (SP)+
20924 123530  000402                          BR      6$
20925 123532  005726                  5$:     TST     (SP)+                           ;RESTORE STACK
20926 123534  005726                          TST     (SP)+
20927 123536  022702  000140          6$:     CMP     #140,R2                         ;PRIORITY 3 LAST?
```

TEST - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS

```
20928 123542  001342                    BNE     2$                    ;IF NOT, BRANCH TO CONTINUE
20929
```

TEST - PMG COUNTER

```
                                .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 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 (CLEARING OF CSR2<0> IS "STOLEN" CPU BUS CYCLE)
                                ;.      IF 2ND DMA DIDN'T HAPPENED THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;ENDROUTINE

                                ;;*****************************************************************
  123544  000004                TST66:  SCOPE
20962 123546  000240                    NOP
20963 123550  005737  003032      TST     CCHPAS                  ;HAVE DONE ENOUGH INCLUSIVE PASSES?
20964 123554  001003              BNE     99$                     ; NOT YET
20965 123556  000240              NOP                             ; DEBUG AID
20966 123560  000137  124264      JMP     11$                     ; YES SKIP THIS
20967 123564  000240          99$:NOP
20968
20969 123566  032737  000200  000052  BIT     #BIT07,@#52                 ;UFD MODE?
20970 123574  001402              BEQ     100$                        ;IF NOT, CONTINUE
20971 123576  000137  135446      JMP     $EOP                        ;EXIT
20972 123602  005737  002704  100$:TST     CSR12                       ;SECOND Q22BE AVAILABLE?
20973 123606  001002              BNE     110$                        ;IF YES, GO DO TEST
20974 123610  000137  135446      JMP     $EOP                        ;OTHERWISE, GOTO EOP
20975                           ;
20976                           ; DETERMINE WHICH Q22BE IS CLOSER TO CPU BY DOING DATO
20977                           ; THE CSR1 OF THE ONE WITH HIGHER PRIORITY IS IN R4, THE SECOND IN R2
20978                           ;
20979 123614  005077  057046  110$:CLR     @CSR2                       ;CLEAR JUST IN CASE
20980 123620  005077  057062      CLR     @CSR22
20981 123624  012777  002740  057036  MOV     #TEMP,@BA                   ;ADDRESS TO BE USED
20982 123632  012777  002740  057050  MOV     #TEMP,@BA2                  ;FOR BOTH OF THEM
20983 123640  012777  177777  057024  MOV     #177777,@WC                 ;DO FOR 1 WORD
20984 123646  012777  177777  057036  MOV     #177777,@WC2                ;IN BOTH Q22BW'S
20985 123654  012777  012525  057012  MOV     #12525,@DATA                ;DATA FOR 1ST
20986 123662  012777  052525  057024  MOV     #52525,@DATA2               ;DATA FOR 2ND
```

TEST - PMG COUNTER

```
20987 123670 012777 001601 056766          MOV     #1601,@CSR1             ;1 DATO FOR 1ST
20988 123676 012777 001601 057000          MOV     #1601,@CSR12            ;AND 2ND
20989 123704 012777 000001 056770          MOV     #1,@SIMGOA              ;BOTH GO
20990 123712 105777 056746          1$:    TSTB    @CSR1                   ;FIRST DONE?
20991 123716 100375                        BPL     1$                      ;IF NOT, WAIT
20992 123720 105777 056760          2$:    TSTB    @CSR12                  ;SECOND DONE
20993 123724 100375                        BPL     2$                      ;WAIT FOR 2ND
20994 123726 022737 052525 002740          CMP     #52525,TEMP             ;SECOND FINISHED LAST?
20995 123734 001405                        BEQ     3$                      ;IF SO, BRANCH
20996 123736 013704 002704                 MOV     CSR12,R4                ;SECOND ONE AT HIGHER LEVEL
20997 123742 013702 002664                 MOV     CSR1,R2                 ;FIRST AT LOWER
20998 123746 000404                        BR      4$                      ;DO PMG PART
20999 123750 013704 002664          3$:    MOV     CSR1,R4                 ;FIRST ONE AT HIGHER LEVEL
21000 123754 013702 002704                 MOV     CSR12,R2                ;SECOND AT LOWER
21001                               ;
21002                               ; INITIATE DMA CYCLES TO WORK WITH PMG COUNTER
21003                               ;
21004 123760 013737 177520 002730   4$:    MOV     BCSR,SAVBR              ;STORE BCSR REGISTER
21005 123766 012700 000001                 MOV     #BIT00,R0               ;FIRST VALUE FOR PMG 0.4MSEC
21006 123772 050037 177520          5$:    BIS     R0,BCSR                 ;CHANGE PMG CONUTER
21007 123776 005737 124100                 TST     6$                      ;CACHE RESET INTRUCTION
21008 124002 005737 124102                 TST     6$+2                    ;AND THE FOLLOWING FEW
21009 124006 005737 124104                 TST     6$+4                    ;
21010 124012 005737 124106                 TST     6$+6                    ;
21011 124016 012714 001407                 MOV     #1407,(R4)              ;DATI IN HOG MODE FOR HIGHER ONE
21012 124022 005064 000002                 CLR     2(R4)                   ;CLEAR CSR2
21013 124026 012764 002740 000004          MOV     #TEMP,4(R4)             ;ADDRESS TO START DATI
21014 124034 012764 000000 000006          MOV     #0,6(R4)                ;DO 128 DATI'S IN HOG MODE
21015 124042 012712 001407                 MOV     #1407,(R2)              ;DATO FOR LOWER LEVEL ONE
21016 124046 005062 000002                 CLR     2(R2)                   ;CLEAR JUST IN CASE
21017 124052 012762 002740 000004          MOV     #TEMP,4(R2)             ;USE THE SAME ADDRESS
21018 124060 012762 177777 000006          MOV     #177777,6(R2)           ;DO JUST ONE DATO
21019 124066 005062 000010                 CLR     10(R2)                  ;CLEAR DATA OF 2ND Q22BE
21020 124072 012762 000001 000016          MOV     #1,16(R2)               ;BOTH GO
21021 124100 000005                 6$:    RESET                           ;STOP Q22BE AT R4
21022 124102 023762 002740 000010          CMP     TEMP,10(R2)             ;SECOND DMA DONE?
21023 124110 001001                        BNE     7$                      ;IF SET, BRANCH
21024 124112 104127                        ERROR   +127                    ;NO CYCLE STEALING
21025 124114 062700 000003          7$:    ADD     #3,R0                   ;DO FOR 1,3,7 IN PMG
21026 124120 040037 177520                 BIC     R0,BCSR                 ;CLEAR PREVOIUS BITS IN BCSR
21027 124124 022700 000007                 CMP     #7,R0                   ;LAST ONE?
21028 124130 002320                        BGE     5$                      ;IF NOT, BRANCH
21029                               ;
21030                               ; TRY WITHOUT PMG COUNTER OPERATING
21031                               ;
21032 124132 042737 000007 177520          BIC     #7,BCSR                 ;TURN OF PMG COUNTER
21033 124140 005737 124242                 TST     8$                      ;CACHE RESET FETCH
21034 124144 005737 124244                 TST     8$+2                    ;AND THE FOLLOWING FEW
21035 124150 005737 124246                 TST     8$+4                    ;
21036 124154 005737 124250                 TST     8$+6                    ;
21037 124160 012714 001407                 MOV     #1407,(R4)              ;DATI IN HOG MODE FOR HIGHER ONE
21038 124164 005064 000002                 CLR     2(R4)                   ;CLEAR CSR2
21039 124170 012764 002740 000004          MOV     #TEMP,4(R4)             ;ADDRESS TO START DATI
21040 124176 012764 000000 000006          MOV     #0,6(R4)                ;DO 128 DATI'S IN HOG MODE
21041 124204 012712 001407                 MOV     #1407,(R2)              ;DATO FOR LOWER LEVEL ONE
21042 124210 005062 000002                 CLR     2(R2)                   ;CLEAR CRS2 OF 2ND
21043 124214 012762 002740 000004          MOV     #TEMP,4(R2)             ;USE THE SAME ADDRESS
```

TEST - PMG COUNTER

```
21044 124222  012762  177777  000006          MOV     #177777,6(R2)                   ;DO JUST ONE DATO
21045 124230  005062  000010                  CLR     10(R2)                          ;CLEAR DATA OF 2ND Q22BE
21046 124234  012777  000001  056440          MOV     #1,@SIMGOA                      ;BOTH GO
21047 124242  000005                  8$:     RESET                                   ;IF NOT WORKING, STOPS 2ND
21048 124244  023762  002740  000010          CMP     TEMP,10(R2)                     ;2ND DMA HAPPENED?
21049 124252  001401                          BEQ     9$                              ;IF YES, BRANCH
21050 124254  104127                          ERROR   +127                            ;IN PMG COUNTER
21051 124256  013737  002730  177520  9$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
21052 124264  000240                  11$:    NOP
21053 124266  000137  135446                  JMP     $EOP
21054
21055 124272  123727  001220  000001  VIREOP: CMPB    $ENV,#1                         ; IF NOT APT, DON'T WORRY ABOUT
21056 124300  001005                          BNE     1$                              ;
21057 124302  005737  003032                  TST     CCHPAS                          ; MAINTAIN CACHE ROUTIN PASCNT
21058 124306  001402                          BEQ     1$
21059 124310  005337  003032                  DEC     CCHPAS
21060
21061                                         ; THIS VIREOP ROUTINE TO PROVIDE COMMON END OF PASS EXIT POINT
21062 124314  000205                  1$:     RTS     R5
21063
```

```
21065                                   .SBTTL  GLOBAL ERROR MESSAGES
21066 124316    102     101     123  EM1:    .ASCIZ  /BASIC INSTRUCTION SET ERROR/
      124321    111     103     040
      124324    111     116     123
      124327    124     122     125
      124332    103     124     111
      124335    117     116     040
      124340    123     105     124
      124343    040     105     122
      124346    122     117     122
      124351    000
21067 124352    115     115     125  EM2:    .ASCIZ  /MMU ERROR/
      124355    040     105     122
      124360    122     117     122
      124363    000
21068 124364    106     120     120  EM3:    .ASCIZ  /FPP ERROR/
      124367    040     105     122
      124372    122     117     122
      124375    000
21069 124376    105     122     122  EM4:    .ASCIZ  /ERROR IN READ-WRITE BITS OF CCR/
      124401    117     122     040
      124404    111     116     040
      124407    122     105     101
      124412    104     055     127
      124415    122     111     124
      124420    105     040     102
      124423    111     124     123
      124426    040     117     106
      124431    040     103     103
      124434    122     000
21070 124436    106     117     122  EM5:    .ASCIZ  /FORCE MISS WRITES TO CACHE/
      124441    103     105     040
      124444    115     111     123
      124447    123     040     127
      124452    122     111     124
      124455    105     123     040
      124460    124     117     040
      124463    103     101     103
      124466    110     105     000
21071 124471    106     117     122  EM6:    .ASCIZ  /FORCE MISS WRITE INVALIDATES CACHE/
      124474    103     105     040
      124477    115     111     123
      124502    123     040     127
      124505    122     111     124
      124510    105     040     111
      124513    116     126     101
      124516    114     111     104
      124521    101     124     105
      124524    123     040     103
      124527    101     103     110
      124532    105     000
21072 124534    125     116     105  EM7:    .ASCIZ  /UNEXPECTED PARITY INTERRUPT/
      124537    130     120     105
      124542    103     124     105
      124545    104     040     120
      124550    101     122     111
      124553    124     131     040
```

GLOBAL ERROR MESSAGES

```
                124556  111     116     124
                124561  105     122     122
                124564  125     120     124
                124567  000
21073           124570  124     101     107     EM10:   .ASCIZ  /TAG PARITY ERROR/
                124573  040     120     101
                124576  122     111     124
                124601  131     040     105
                124604  122     122     117
                124607  122     000
21074           124611  104     101     124     EM11:   .ASCIZ  /DATA PARITY ERROR/
                124614  101     040     120
                124617  101     122     111
                124622  124     131     040
                124625  105     122     122
                124630  117     122     000
21075           124633  114     117     127     EM12:   .ASCIZ  /LOW BYTE PARITY ERROR/
                124636  040     102     131
                124641  124     105     040
                124644  120     101     122
                124647  111     124     131
                124652  040     105     122
                124655  122     117     122
                124660  000
21076           124661  110     111     107     EM13:   .ASCIZ  /HIGH BYTE PARITY ERROR/
                124664  110     040     102
                124667  131     124     105
                124672  040     120     101
                124675  122     111     124
                124700  131     040     105
                124703  122     122     117
                124706  122     000
21077           124710  105     122     122     EM14:   .ASCIZ  /ERROR IN DATA PATH/
                124713  117     122     040
                124716  111     116     040
                124721  104     101     124
                124724  101     040     120
                124727  101     124     110
                124732  000
21078           124733  106     117     122     EM15:   .ASCIZ  /FORCE MISS READS FROM CACHE/
                124736  103     105     040
                124741  115     111     123
                124744  123     040     122
                124747  105     101     104
                124752  123     040     106
                124755  122     117     115
                124760  040     103     101
                124763  103     110     105
                124766  000
21079           124767  106     117     122     EM16:   .ASCIZ  /FORCE MISS READS FROM CACHE AND MISS/
                124772  103     105     040
                124775  115     111     123
                125000  123     040     122
                125003  105     101     104
                125006  123     040     106
                125011  122     117     115
                125014  040     103     101
```

GLOBAL ERROR MESSAGES

```
        125017  103  110  105
        125022  040  101  116
        125025  104  040  115
        125030  111  123  123
        125033  000
21080   125034  105  122  122  EM17:   .ASCIZ  \ERROR IN RECORDING HITS IN HIT/MISS\
        125037  117  122  040
        125042  111  116  040
        125045  122  105  103
        125050  117  122  104
        125053  111  116  107
        125056  040  110  111
        125061  124  123  040
        125064  111  116  040
        125067  110  111  124
        125072  057  115  111
        125075  123  123  000
21081   125100  127  122  111  EM20:   .ASCIZ  /WRITE BYTE ALLOCATES CACHE/
        125103  124  105  040
        125106  102  131  124
        125111  105  040  101
        125114  114  114  117
        125117  103  101  124
        125122  105  123  040
        125125  103  101  103
        125130  110  105  000
21082   125133  127  122  111  EM21:   .ASCIZ  /WRITE BYTE HIT DOES NOT RECORD HIT/
        125136  124  105  040
        125141  102  131  124
        125144  105  040  110
        125147  111  124  040
        125152  104  117  105
        125155  123  040  116
        125160  117  124  040
        125163  122  105  103
        125166  117  122  104
        125171  040  110  111
        125174  124  000
21083   125176  102  131  124  EM22:   .ASCIZ  /BYTES REVERSED ON WRITE CYCLES/
        125201  105  123  040
        125204  122  105  126
        125207  105  122  123
        125212  105  104  040
        125215  117  116  040
        125220  127  122  111
        125223  124  105  040
        125226  103  131  103
        125231  114  105  123
        125234  000
21084   125235  103  117  116  EM23:   .ASCIZ  /CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE/
        125240  104  111  124
        125243  111  117  116
        125246  101  114  040
        125251  102  131  120
        125254  101  123  123
        125257  040  104  117
        125262  105  123  116
```

GLOBAL ERROR MESSAGES

```
                125265  047     124     040
                125270  111     116     126
                125273  101     114     111
                125276  104     101     124
                125301  105     040     103
                125304  101     103     110
                125307  105     000
21085           125311  110     111     124     EM24:   .ASCIZ  /HITS RECORDED AFTER FLUSHING CACHE/
                125314  123     040     122
                125317  105     103     117
                125322  122     104     105
                125325  104     040     101
                125330  106     124     105
                125333  122     040     106
                125336  114     125     123
                125341  110     111     116
                125344  107     040     103
                125347  101     103     110
                125352  105     000
21086           125354  102     131     120     EM25:   .ASCIZ  /BYPASS DOESN'T INVALIDATE CACHE/
                125357  101     123     123
                125362  040     104     117
                125365  105     123     116
                125370  047     124     040
                125373  111     116     126
                125376  101     114     111
                125401  104     101     124
                125404  105     040     103
                125407  101     103     110
                125412  105     000
21087           125414  115     123     105     EM26:   .ASCIZ  /MSER DOES NOT CLEAR ON WRITE REFERENCE/
                125417  122     040     104
                125422  117     105     123
                125425  040     116     117
                125430  124     040     103
                125433  114     105     101
                125436  122     040     117
                125441  116     040     127
                125444  122     111     124
                125447  105     040     122
                125452  105     106     105
                125455  122     105     116
                125460  103     105     000
21088           125463  120     101     122     EM27:   .ASCIZ  /PARITY ERROR DON'T CAUSE A MISS/
                125466  111     124     131
                125471  040     105     122
                125474  122     117     122
                125477  040     104     117
                125502  116     047     124
                125505  040     103     101
                125510  125     123     105
                125513  040     101     040
                125516  115     111     123
                125521  123     000
21089           125523  120     101     122     EM30:   .ASCIZ  /PARITY ERROR DON'T SET MSER WITH CCR<7>=0/
                125526  111     124     131
                125531  040     105     122
```

GLOBAL ERROR MESSAGES

```
                125534  122     117     122
                125537  040     104     117
                125542  116     047     124
                125545  040     123     105
                125550  124     040     115
                125553  123     105     122
                125556  040     127     111
                125561  124     110     040
                125564  103     103     122
                125567  074     067     076
                125572  075     060     000
21090           125575  120     101     122     EM31:   .ASCIZ  /PARITY ERROR IGNORED/
                125600  111     124     131
                125603  040     105     122
                125606  122     117     122
                125611  040     111     107
                125614  116     117     122
                125617  105     104     000
21091           125622  120     101     122     EM32:   .ASCIZ  /PARITY ERROR IGNORED ON LOW BYTE/
                125625  111     124     131
                125630  040     105     122
                125633  122     117     122
                125636  040     111     107
                125641  116     117     122
                125644  105     104     040
                125647  117     116     040
                125652  114     117     127
                125655  040     102     131
                125660  124     105     000
21092           125663  120     101     122     EM33:   .ASCIZ  /PARITY ERROR IGNORED ON HIGH BYTE/
                125666  111     124     131
                125671  040     105     122
                125674  122     117     122
                125677  040     111     107
                125702  116     117     122
                125705  105     104     040
                125710  117     116     040
                125713  110     111     107
                125716  110     040     102
                125721  131     124     105
                125724  000
21093           125725  120     101     122     EM34:   .ASCIZ  /PARITY ABORT LOGIC DOESN'T WORK/
                125730  111     124     131
                125733  040     101     102
                125736  117     122     124
                125741  040     114     117
                125744  107     111     103
                125747  040     104     117
                125752  105     123     116
                125755  047     124     040
                125760  127     117     122
                125763  113     000
21094           125765  115     123     105     EM35:   .ASCIZ  /MSER NOT SET PROPERLY/
                125770  122     040     116
                125773  117     124     040
                125776  123     105     124
                126001  040     120     122
```

GLOBAL ERROR MESSAGES

```
                126004  117     120     105
                126007  122     114     131
                126012  000
21095           126013  120     101     122     EM36:   .ASCIZ  /PARITY INTERRUPT LOGIC DOESN'T WORK/
                126016  111     124     131
                126021  040     111     116
                126024  124     105     122
                126027  122     125     120
                126032  124     040     114
                126035  117     107     111
                126040  103     040     104
                126043  117     105     123
                126046  116     047     124
                126051  040     127     117
                126054  122     113     000
21096           126057  116     130     115     EM37:   .ASCIZ  /NXM AND PARITY ABORT DIN'T HAPPEN/
                126062  040     101     116
                126065  104     040     120
                126070  101     122     111
                126073  124     131     040
                126076  101     102     117
                126101  122     124     040
                126104  104     111     116
                126107  047     124     040
                126112  110     101     120
                126115  120     105     116
                126120  000
21097           126121  120     101     122     EM40:   .ASCIZ  /PARITY ABORT NOT BLOCKED BY NXM TRAP/
                126124  111     124     131
                126127  040     101     102
                126132  117     122     124
                126135  040     116     117
                126140  124     040     102
                126143  114     117     103
                126146  113     105     104
                126151  040     102     131
                126154  040     116     130
                126157  115     040     124
                126162  122     101     120
                126165  000
21098           126166  115     125     114     EM41:   .ASCIZ  /MULTI-PROCESSOR HOUK INSTRUCTION DOESN'T CAUSE MISS/
                126171  124     111     055
                126174  120     122     117
                126177  103     105     123
                126202  123     117     122
                126205  040     110     117
                126210  117     113     040
                126213  111     116     123
                126216  124     122     125
                126221  103     124     111
                126224  117     116     040
                126227  104     117     105
                126232  123     116     047
                126235  124     040     103
                126240  101     125     123
                126243  105     040     115
                126246  111     123     123
```

GLOBAL ERROR MESSAGES

```
                126251  000
21099           126252  105     122     122     EM42:   .ASCIZ  /ERROR IN PARITY LOGIC/
                126255  117     122     040
                126260  111     116     040
                126263  120     101     122
                126266  111     124     131
                126271  040     114     117
                126274  107     111     103
                126277  000
21100           126300  105     122     122     EM43:   .ASCIZ  /ERROR IN CACHE DATA RAMS/
                126303  117     122     040
                126306  111     116     040
                126311  103     101     103
                126314  110     105     040
                126317  104     101     124
                126322  101     040     122
                126325  101     115     123
                126330  000
21101           126331  105     122     122     EM44:   .ASCIZ  /ERROR IN NXM IN STANDALONE MODE/
                126334  117     122     040
                126337  111     116     040
                126342  116     130     115
                126345  040     111     116
                126350  040     123     124
                126353  101     116     104
                126356  101     114     117
                126361  116     105     040
                126364  115     117     104
                126367  105     000
21102           126371  105     122     122     EM45:   .ASCIZ  \ERROR IN RECORDING HITS THROUGH HIT/MISS REGISTER\
                126374  117     122     040
                126377  111     116     040
                126402  122     105     103
                126405  117     122     104
                126410  111     116     107
                126413  040     110     111
                126416  124     123     040
                126421  124     110     122
                126424  117     125     107
                126427  110     040     110
                126432  111     124     057
                126435  115     111     123
                126440  123     040     122
                126443  105     107     111
                126446  123     124     105
                126451  122     000
21103           126453  110     111     124     EM46:   .ASCIZ  /HIT RECORDED FOR A LOCATION THAT SHOULD NOT BE IN CACHE/
                126456  040     122     105
                126461  103     117     122
                126464  104     105     104
                126467  040     106     117
                126472  122     040     101
                126475  040     114     117
                126500  103     101     124
                126503  111     117     116
                126506  040     124     110
                126511  101     124     040
```

GLOBAL ERROR MESSAGES

```
        126514  123  110  117
        126517  125  114  104
        126522  040  116  117
        126525  124  040  102
        126530  105  040  111
        126533  116  040  103
        126536  101  103  110
        126541  105  000
 21104  126543  115  111  123  EM47:   .ASCIZ  /MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE/
        126546  123  040  122
        126551  105  103  117
        126554  122  104  105
        126557  104  040  106
        126562  117  122  040
        126565  101  040  114
        126570  117  103  101
        126573  124  111  117
        126576  116  040  124
        126601  110  101  124
        126604  040  123  110
        126607  117  125  114
        126612  104  040  102
        126615  105  040  111
        126620  116  040  103
        126623  101  103  110
        126626  105  000
 21105  126630  105  122  122  EM50:   .ASCIZ  /ERROR IN TAG STORE/
        126633  117  122  040
        126636  111  116  040
        126641  124  101  107
        126644  040  123  124
        126647  117  122  105
        126652  000
 21106  126653  105  122  122  EM51:   .ASCIZ  /ERROR PCR READ-WRITE BITS/
        126656  117  122  040
        126661  120  103  122
        126664  040  122  105
        126667  101  104  055
        126672  127  122  111
        126675  124  105  040
        126700  102  111  124
        126703  123  000
 21107  126705  105  122  122  EM52:   .ASCIZ  /ERROR IN BCSR READ-WRITE BITS/
        126710  117  122  040
        126713  111  116  040
        126716  102  103  123
        126721  122  040  122
        126724  105  101  104
        126727  055  127  122
        126732  111  124  105
        126735  040  102  111
        126740  124  123  000
 21108  126743  122  105  123  EM53:   .ASCIZ  /RESET DOESN'T CLEAR BCSR<4>/
        126746  105  124  040
        126751  104  117  105
        126754  123  116  047
        126757  124  040  103
```

GLOBAL ERROR MESSAGES

```
                126762  114     105     101
                126765  122     040     102
                126770  103     123     122
                126773  074     064     076
                126776  000
21109           126777  103     110     105     EM54:   .ASCIZ  /CHECKSUM ERROR IN 16-BIT ROM /
                127002  103     113     123
                127005  125     115     040
                127010  105     122     122
                127013  117     122     040
                127016  111     116     040
                127021  061     066     055
                127024  102     111     124
                127027  040     122     117
                127032  115     040     000
21110           127035  103     110     105     EM55:   .ASCIZ  /CHECKSUM ERROR IN 8-BIT ROM/
                127040  103     113     123
                127043  125     115     040
                127046  105     122     122
                127051  117     122     040
                127054  111     116     040
                127057  070     055     102
                127062  111     124     040
                127065  122     117     115
                127070  000
21111           127071  124     111     115     EM56:   .ASCIZ  /TIMEOUT READING LKS/
                127074  105     117     125
                127077  124     040     122
                127102  105     101     104
                127105  111     116     107
                127110  040     114     113
                127113  123     000
21112           127115  114     113     123     EM57:   .ASCIZ  /LKS<07> DOES NOT BECOME 1/
                127120  074     060     067
                127123  076     040     104
                127126  117     105     123
                127131  040     116     117
                127134  124     040     102
                127137  105     103     117
                127142  115     105     040
                127145  061     000
21113           127147  127     122     111     EM60:   .ASCIZ  /WRITE REFERENCE DOESN'T CLEAR LKS<07>/
                127152  124     105     040
                127155  122     105     106
                127160  105     122     105
                127163  116     103     105
                127166  040     104     117
                127171  105     123     116
                127174  047     124     040
                127177  103     114     105
                127202  101     122     040
                127205  114     113     123
                127210  074     060     067
                127213  076     000
21114           127215  111     114     114     EM61:   .ASCIZ  /ILLEGAL LKS INTERRUPTS/
                127220  105     107     101
                127223  114     040     114
```

GLOBAL ERROR MESSAGES

```
                127226  113  123  040
                127231  111  116  124
                127234  105  122  122
                127237  125  120  124
                127242  123  000
21115           127244  120  122  117  EM62:   .ASCIZ  /PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>/
                127247  103  105  123
                127252  123  117  122
                127255  040  111  116
                127260  124  105  122
                127263  122  125  120
                127266  124  123  040
                127271  104  117  116
                127274  047  124  040
                127277  103  114  105
                127302  101  122  040
                127305  114  113  123
                127310  074  060  067
                127313  076  000
21116           127315  114  113  123  EM63:   .ASCIZ  /LKS READY DOESN'T GO LOW/
                127320  040  122  105
                127323  101  104  131
                127326  040  104  117
                127331  105  123  116
                127334  047  124  040
                127337  107  117  040
                127342  114  117  127
                127345  000
21117           127346  127  122  117  EM64:   .ASCIZ  /WRONG NUMBER OF LKS INTERRUPTS/
                127351  116  107  040
                127354  116  125  115
                127357  102  105  122
                127362  040  117  106
                127365  040  114  113
                127370  123  040  111
                127373  116  124  105
                127376  122  122  125
                127401  120  124  123
                127404  000
21118           127405  114  113  123  EM65:   .ASCIZ  /LKS INTERRUPTS HAPPEN AT WRONG PRIORITY/
                127410  040  111  116
                127413  124  105  122
                127416  122  125  120
                127421  124  123  040
                127424  110  101  120
                127427  120  105  116
                127432  040  101  124
                127435  040  127  122
                127440  117  116  107
                127443  040  120  122
                127446  111  117  122
                127451  111  124  131
                127454  000
21119           127455  102  103  123  EM66:   .ASCIZ  /BCSR<12> DOES NOT DISABLES LKS/
                127460  122  074  061
                127463  062  076  040
                127466  104  117  105
```

GLOBAL ERROR MESSAGES

```
                127471  123     040     116
                127474  117     124     040
                127477  104     111     123
                127502  101     102     114
                127505  105     123     040
                127510  114     113     123
                127513  000
21120           127514  102     103     123     EM67:   .ASCIZ  /BCSR<13> DOESN'T SET LKS<06>/
                127517  122     074     061
                127522  063     076     040
                127525  104     117     105
                127530  123     116     047
                127533  124     040     123
                127536  105     124     040
                127541  114     113     123
                127544  074     060     066
                127547  076     000
21121           127551  122     105     123     EM70:   .ASCIZ  /RESET DOESN'T SET LKS<7>/
                127554  105     124     040
                127557  104     117     105
                127562  123     116     047
                127565  124     040     123
                127570  105     124     040
                127573  114     113     123
                127576  074     067     076
                127601  000
21122           127602  122     105     123     EM71:   .ASCIZ  /RESET DOESN'T CLEAR LKS<06>/
                127605  105     124     040
                127610  104     117     105
                127613  123     116     047
                127616  124     040     103
                127621  114     105     101
                127624  122     040     114
                127627  113     123     074
                127632  060     066     076
                127635  000
21123           127636  124     111     115     EM72:   .ASCIZ  /TIMEOUT READING SLU REGISTERS/
                127641  105     117     125
                127644  124     040     122
                127647  105     101     104
                127652  111     116     107
                127655  040     123     114
                127660  125     040     122
                127663  105     107     111
                127666  123     124     105
                127671  122     123     000
21124           127674  105     122     122     EM73:   .ASCIZ  /ERROR IN XMIT READY/
                127677  117     122     040
                127702  111     116     040
                127705  130     115     111
                127710  124     040     122
                127713  105     101     104
                127716  131     000
21125           127720  122     103     123     EM74:   .ASCIZ  /RCSR<7> DOESN'T BECOME 1/
                127723  122     074     067
                127726  076     040     104
                127731  117     105     123
```

GLOBAL ERROR MESSAGES

```
        127734  116  047  124
        127737  040  102  105
        127742  103  117  115
        127745  105  040  061
        127750  000
21126   127751  127  122  117  EM75:   .ASCIZ  /WRONG CHARACTER RECEIVED/
        127754  116  107  040
        127757  103  110  101
        127762  122  101  103
        127765  124  105  122
        127770  040  122  105
        127773  103  105  111
        127776  126  105  104
        130001  000
21127   130002  122  103  123  EM76:   .ASCIZ  /RCSR<07> NOT CLEARED AFTER READING RBUF/
        130005  122  074  060
        130010  067  076  040
        130013  116  117  124
        130016  040  103  114
        130021  105  101  122
        130024  105  104  040
        130027  101  106  124
        130032  105  122  040
        130035  122  105  101
        130040  104  111  116
        130043  107  040  122
        130046  102  125  106
        130051  000
21128   130052  130  103  123  EM77:   .ASCIZ  /XCSR<07> NOT SET ON RESET/
        130055  122  074  060
        130060  067  076  040
        130063  116  117  124
        130066  040  123  105
        130071  124  040  117
        130074  116  040  122
        130077  105  123  105
        130102  124  000
21129   130104  122  103  123  EM100:  .ASCIZ  /RCSR<07> NOT CLEARED ON RESET/
        130107  122  074  060
        130112  067  076  040
        130115  116  117  124
        130120  040  103  114
        130123  105  101  122
        130126  105  104  040
        130131  117  116  040
        130134  122  105  123
        130137  105  124  000
21130   130142  123  114  125  EM101:  .ASCIZ  /SLU INTERRUPTS HAPPEN AT 4/
        130145  040  111  116
        130150  124  105  122
        130153  122  125  120
        130156  124  123  040
        130161  110  101  120
        130164  120  105  116
        130167  040  101  124
        130172  040  064  000
21131   130175  122  105  123  EM102:  .ASCIZ  /RESET DOES NOT CLEAR PROPER BITS IN SLU REGISTERS/
```

GLOBAL ERROR MESSAGES

```
        130200      105     124     040
        130203      104     117     105
        130206      123     040     116
        130211      117     124     040
        130214      103     114     105
        130217      101     122     040
        130222      120     122     117
        130225      120     105     122
        130230      040     102     111
        130233      124     123     040
        130236      111     116     040
        130241      123     114     125
        130244      040     122     105
        130247      107     111     123
        130252      124     105     122
        130255      123     000
 21132  130257      124     122     101  EM103:  .ASCIZ  /TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>/
        130262      116     123     115
        130265      111     124     040
        130270      111     116     124
        130273      105     122     122
        130276      125     120     124
        130301      040     104     117
        130304      105     123     040
        130307      116     117     124
        130312      040     103     114
        130315      105     101     122
        130320      040     130     103
        130323      123     122     074
        130326      060     067     076
        130331      000
 21133  130332      122     105     103  EM104:  .ASCIZ  /RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>/
        130335      105     111     126
        130340      105     040     111
        130343      116     124     105
        130346      122     122     125
        130351      120     124     123
        130354      040     104     117
        130357      116     047     124
        130362      040     103     114
        130365      105     101     122
        130370      040     122     103
        130373      123     122     074
        130376      060     067     076
        130401      000
 21134  130402      102     122     105  EM105:  .ASCIZ  /BREAK CONDITION DOES NOT SET RBUF PROPERLY/
        130405      101     113     040
        130410      103     117     116
        130413      104     111     124
        130416      111     117     116
        130421      040     104     117
        130424      105     123     040
        130427      116     117     124
        130432      040     123     105
        130435      124     040     122
        130440      102     125     106
        130443      040     120     122
```

GLOBAL ERROR MESSAGES

```
        130446  117  120  105
        130451  122  114  131
        130454  000
 21135  130455  122  102  125  EM106:  .ASCIZ  /RBUF <15-11> WASN'T CLEARED ON NEXT CHARACTER/
        130460  106  040  074
        130463  061  065  055
        130466  061  061  076
        130471  040  127  101
        130474  123  116  047
        130477  124  040  103
        130502  114  105  101
        130505  122  105  104
        130510  040  117  116
        130513  040  116  105
        130516  130  124  040
        130521  103  110  101
        130524  122  101  103
        130527  124  105  122
        130532  000
 21136  130533  123  114  125  EM107:  .ASCIZ  /SLU INTERRUPTS DON'T HAPPEN/
        130536  040  111  116
        130541  124  105  122
        130544  122  125  120
        130547  124  123  040
        130552  104  117  116
        130555  047  124  040
        130560  110  101  120
        130563  120  105  116
        130566  000
 21137  130567  105  122  122  EM110:  .ASCIZ  /ERROR IN WRITING TO SLU REGISTERS/
        130572  117  122  040
        130575  111  116  040
        130600  127  122  111
        130603  124  111  116
        130606  107  040  124
        130611  117  040  123
        130614  114  125  040
        130617  122  105  107
        130622  111  123  124
        130625  105  122  123
        130630  000
 21138  130631  106  111  122  EM111:  .ASCIZ  /FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND/
        130634  123  124  040
        130637  103  110  101
        130642  122  101  103
        130645  124  105  122
        130650  040  127  101
        130653  123  040  116
        130656  117  124  040
        130661  117  126  105
        130664  122  122  125
        130667  116  040  102
        130672  131  040  124
        130675  110  105  040
        130700  123  105  103
        130703  117  116  104
        130706  000
```

GLOBAL ERROR MESSAGES

```
21139 130707  117  126  105  EM112:  .ASCIZ  /OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF/
      130712  122  122  125
      130715  116  040  103
      130720  117  116  104
      130723  111  124  111
      130726  117  116  040
      130731  104  117  105
      130734  123  040  116
      130737  117  124  040
      130742  123  105  124
      130745  040  120  122
      130750  117  120  105
      130753  122  040  102
      130756  111  124  123
      130761  040  111  116
      130764  040  122  102
      130767  125  106  000
21140 130772  117  126  105  EM113:  .ASCIZ  /OVERRUN BITS WERE NOT CLEARED ON THE NEXT CHARACTER/
      130775  122  122  125
      131000  116  040  102
      131003  111  124  123
      131006  040  127  105
      131011  122  105  040
      131014  116  117  124
      131017  040  103  114
      131022  105  101  122
      131025  105  104  040
      131030  117  116  040
      131033  124  110  105
      131036  040  116  105
      131041  130  124  040
      131044  103  110  101
      131047  122  101  103
      131052  124  105  122
      131055  000
21141 131056  105  122  122  EM114:  .ASCIZ  \ERROR ON XCSR<2>\
      131061  117  122  040
      131064  117  116  040
      131067  130  103  123
      131072  122  074  062
      131075  076  000
21142 131077  105  122  122  EM115:  .ASCIZ  /ERROR IN TAG STORE FROM STANDALONE MODE/
      131102  117  122  040
      131105  111  116  040
      131110  124  101  107
      131113  040  123  124
      131116  117  122  105
      131121  040  106  122
      131124  117  115  040
      131127  123  124  101
      131132  116  104  101
      131135  114  117  116
      131140  105  040  115
      131143  117  104  105
      131146  000
21143 131147  104  115  101  EM116:  .ASCIZ  /DMA TAG PARITY DOES NOT GENERATE PROPER RESPONSE/
      131152  040  124  101
```

GLOBAL ERROR MESSAGES

```
        131155  107  040  120
        131160  101  122  111
        131163  124  131  040
        131166  104  117  105
        131171  123  040  116
        131174  117  124  040
        131177  107  105  116
        131202  105  122  101
        131205  124  105  040
        131210  120  122  117
        131213  120  105  122
        131216  040  122  105
        131221  123  120  117
        131224  116  123  105
        131227  000
21144   131230  115  123  105  EM117:  .ASCIZ  /MSER<13>NOT SET IN STANDALONE MODE/
        131233  122  074  061
        131236  063  076  116
        131241  117  124  040
        131244  123  105  124
        131247  040  111  116
        131252  040  123  124
        131255  101  116  104
        131260  101  114  117
        131263  116  105  040
        131266  115  117  104
        131271  105  000
21145   131273  104  115  101  EM120:  .ASCIZ  /DMA WRITE HITS DON'T INVALIDATE CACHE/
        131276  040  127  122
        131301  111  124  105
        131304  040  110  111
        131307  124  123  040
        131312  104  117  116
        131315  047  124  040
        131320  111  116  126
        131323  101  114  111
        131326  104  101  124
        131331  105  040  103
        131334  101  103  110
        131337  105  000
21146   131341  111  116  040  EM121:  .ASCIZ  /IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED/
        131344  102  114  117
        131347  103  113  040
        131352  115  117  104
        131355  105  040  117
        131360  116  040  127
        131363  122  111  124
        131366  105  040  104
        131371  115  101  040
        131374  110  111  124
        131377  123  040  116
        131402  117  124  040
        131405  105  126  105
        131410  122  131  124
        131413  110  111  116
        131416  107  040  111
        131421  123  040  111
```

GLOBAL ERROR MESSAGES

```
                131424  116  126  101
                131427  114  111  104
                131432  101  124  105
                131435  104  000
21147           131437  122  105  101  EM122:  .ASCIZ  /READ DMA HIT IS WRONG/
                131442  104  040  104
                131445  115  101  040
                131450  110  111  124
                131453  040  111  123
                131456  040  127  122
                131461  117  116  107
                131464  000
21148           131465  105  122  122  EM123:  .ASCIZ  /ERROR IN Q22BE DMA CYCLES/
                131470  117  122  040
                131473  111  116  040
                131476  121  062  062
                131501  102  105  040
                131504  104  115  101
                131507  040  103  131
                131512  103  114  105
                131515  123  000
21149           131517  120  111  122  EM124:  .ASCIZ  /PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS/
                131522  121  040  111
                131525  116  124  105
                131530  122  122  125
                131533  120  124  123
                131536  040  104  117
                131541  116  047  124
                131544  040  124  101
                131547  113  105  040
                131552  120  122  111
                131555  117  122  111
                131560  124  131  040
                131563  117  126  105
                131566  122  040  121
                131571  040  102  125
                131574  123  040  111
                131577  116  124  105
                131602  122  122  125
                131605  120  124  123
                131610  000
21150           131611  116  117  040  EM125:  .ASCIZ  /NO POWER DOWN TRAP TO 24 OCCUR/
                131614  120  117  127
                131617  105  122  040
                131622  104  117  127
                131625  116  040  124
                131630  122  101  120
                131633  040  124  117
                131636  040  062  064
                131641  040  117  103
                131644  103  125  122
                131647  000
21151           131650  105  122  122  EM126:  .ASCIZ  /ERROR DOING Q22BE INTERRUPTS/
                131653  117  122  040
                131656  104  117  111
                131661  116  107  040
                131664  121  062  062
```

GLOBAL ERROR MESSAGES

```
                131667  102  105  040
                131672  111  116  124
                131675  105  122  122
                131700  125  120  124
                131703  123  000
21152           131705  105  122  122  EM127:  .ASCIZ  /ERROR IN OPERATION OF PMG COUNTER/
                131710  117  122  040
                131713  111  116  040
                131716  117  120  105
                131721  122  101  124
                131724  111  117  116
                131727  040  117  106
                131732  040  120  115
                131735  107  040  103
                131740  117  125  116
                131743  124  105  122
                131746  000
21153           131747  125  116  105  EM130:  .ASCIZ  /UNEXPECTED TRAP TO 4/
                131752  130  120  105
                131755  103  124  105
                131760  104  040  124
                131763  122  101  120
                131766  040  124  117
                131771  040  064  000
21154           131774  105  122  122  EM131:  .ASCIZ  /ERROR WRITING TO LKS<6>/
                131777  117  122  040
                132002  127  122  111
                132005  124  111  116
                132010  107  040  124
                132013  117  040  114
                132016  113  123  074
                132021  066  076  000
21155           132024  105  122  122  EM132:  .ASCIZ  /ERROR IN MAINTENANCE REGISTER/
                132027  117  122  040
                132032  111  116  040
                132035  115  101  111
                132040  116  124  105
                132043  116  101  116
                132046  103  105  040
                132051  122  105  107
                132054  111  123  124
                132057  105  122  000
21156           132062  105  105  122  EM133:  .ASCIZ  /EEROM TYPE ERROR/
                132065  117  115  040
                132070  124  131  120
                132073  105  040  105
                132076  122  122  117
                132101  122  000
21157           132103  122  105  101  EM134:  .ASCIZ  /READ OR WRITE EEROM ERROR/
                132106  104  040  117
                132111  122  040  127
                132114  122  111  124
                132117  105  040  105
                132122  105  122  117
                132125  115  040  105
                132130  122  122  117
                132133  122  000
```

GLOBAL ERROR MESSAGES

```
21158
21159 132135  040  124  105  DH1:   .ASCII  / TEST  ERROR/<15><12>
      132140  123  124  011
      132143  105  122  122
      132146  117  122  015
      132151  012
21160 132152  040  040  043         .ASCIZ  /  #     PC/
      132155  011  040  120
      132160  103  000
21161 132162  040  124  105  DH4:   .ASCII  / TEST  ERROR   EXPCTED RECEIVED/<15><12>
      132165  123  124  011
      132170  105  122  122
      132173  117  122  011
      132176  105  130  120
      132201  103  124  105
      132204  104  011  122
      132207  105  103  105
      132212  111  126  105
      132215  104  015  012
21162 132220  040  040  043         .ASCIZ  /  #     PC      DATA     DATA/
      132223  011  040  120
      132226  103  011  040
      132231  104  101  124
      132234  101  040  040
      132237  040  040  040
      132242  104  101  124
      132245  101  000
21163 132247  040  124  105  DH5:   .ASCII  / TEST  ERROR   HITMIS  DATA IN DATA IN/<15><12>
      132252  123  124  011
      132255  105  122  122
      132260  117  122  011
      132263  110  111  124
      132266  115  111  123
      132271  011  104  101
      132274  124  101  040
      132277  111  116  011
      132302  104  101  124
      132305  101  040  111
      132310  116  015  012
21164 132313  040  040  043         .ASCIZ  /  #     PC      REG.   CACHE   MEMORY/
      132316  011  040  120
      132321  103  011  040
      132324  122  105  107
      132327  056  011  103
      132332  101  103  110
      132335  105  011  115
      132340  105  115  117
      132343  122  131  000
21165 132346  040  124  105  DH7:   .ASCII  / TEST  ERROR   ADDRESS MSER/<15><12>
      132351  123  124  011
      132354  105  122  122
      132357  117  122  011
      132362  101  104  104
      132365  122  105  123
      132370  123  011  115
      132373  123  105  122
      132376  015  012
```

F1

GLOBAL ERROR MESSAGES

```
 21166 132400  040  040  043        .ASCIZ  /  #     PC     ACCESSED/
       132403  011  040  120
       132406  103  011  101
       132411  103  103  105
       132414  123  123  105
       132417  104  000
 21167 132421  040  124  105  DH24:  .ASCII  / TEST  ERROR   NUMBER/<15><12>
       132424  123  124  011
       132427  105  122  122
       132432  117  122  011
       132435  116  125  115
       132440  102  105  122
       132443  015  012
 21168 132445  040  040  043        .ASCIZ  /  #     PC     OF HITS/
       132450  011  040  120
       132453  103  011  117
       132456  106  040  110
       132461  111  124  123
       132464  000
 21169 132465  105  122  122  DH27:  .ASCII  \ERROR  ERROR   MSER    HIT/MISS\<15><12>
       132470  117  122  011
       132473  105  122  122
       132476  117  122  011
       132501  115  123  105
       132504  122  011  110
       132507  111  124  057
       132512  115  111  123
       132515  123  015  012
 21170 132520  040  040  043        .ASCIZ  /  #     PC/
       132523  011  040  120
       132526  103  000
 21171 132530  040  124  105  DH41:  .ASCII  / TEST  ERROR   INSTRUCTION/<15><12>
       132533  123  124  011
       132536  105  122  122
       132541  117  122  011
       132544  111  116  123
       132547  124  122  125
       132552  103  124  111
       132555  117  116  015
       132560  012
 21172 132561  040  040  043        .ASCIZ  /  #     PC      OPCODE/
       132564  011  040  120
       132567  103  011  040
       132572  117  120  103
       132575  117  104  105
       132600  000
 21173 132601  040  124  105  DH43:  .ASCII  / TEST  ERROR   EXPECTD RECEIVD CACHE/<15><12>
       132604  123  124  011
       132607  105  122  122
       132612  117  122  011
       132615  105  130  120
       132620  105  103  124
       132623  104  011  122
       132626  105  103  105
       132631  111  126  104
       132634  011  103  101
       132637  103  110  105
```

GLOBAL ERROR MESSAGES

```
                132642  015     012
21174           132644  040     040     043             .ASCIZ  / #     PC      DATA    DATA    LOCATION/
                132647  011     040     120
                132652  103     011     040
                132655  104     101     124
                132660  101     011     040
                132663  104     101     124
                132666  101     011     114
                132671  117     103     101
                132674  124     111     117
                132677  116     000
21175           132701  040     124     105     DH47:   .ASCII  / TEST  ERROR   ADDRESS ADDRESS/<15><12>
                132704  123     124     011
                132707  105     122     122
                132712  117     122     011
                132715  101     104     104
                132720  122     105     123
                132723  123     011     101
                132726  104     104     122
                132731  105     123     123
                132734  015     012
21176           132736  040     040     043             .ASCIZ  / #     PC      <21-16>  <15-0>/
                132741  011     040     120
                132744  103     011     074
                132747  062     061     055
                132752  061     066     076
                132755  011     040     074
                132760  061     065     055
                132763  060     076     000
21177           132766  040     124     105     DH65:   .ASCII  / TEST  ERROR   PRIORITY/<15><12>
                132771  123     124     011
                132774  105     122     122
                132777  117     122     011
                133002  120     122     111
                133005  117     122     111
                133010  124     131     015
                133013  012
21178           133014  040     040     043             .ASCIZ  / #     PC      LEVEL/
                133017  011     040     120
                133022  103     011     114
                133025  105     126     105
                133030  114     000
21179           133032  040     124     105     DH72:   .ASCII  / TEST  ERROR   ADDRESS/<15><12>
                133035  123     124     011
                133040  105     122     122
                133043  117     122     011
                133046  101     104     104
                133051  122     105     123
                133054  123     015     012
21180           133057  040     040     043             .ASCIZ  / #     PC      FAILED/
                133062  011     040     120
                133065  103     011     106
                133070  101     111     114
                133073  105     104     000
21181           133076  040     124     105     DH105:  .ASCII  / TEST  ERROR   RBUF/<15><12>
                133101  123     124     011
                133104  105     122     122
```

GLOBAL ERROR MESSAGES

```
                133107  117  122  011
                133112  122  102  125
                133115  106  015  012
     21182      133120  040  040  043          .ASCIZ  /  #     PC/
                133123  011  040  120
                133126  103  000
     21183      133130  040  124  105  DH115:  .ASCII  / TEST  ERROR   MSER    ADDRESS/<15><12>
                133133  123  124  011
                133136  105  122  122
                133141  117  122  011
                133144  115  123  105
                133147  122  011  101
                133152  104  104  122
                133155  105  123  123
                133160  015  012
     21184      133162  040  040  043          .ASCIZ  /  #     PC              ACCESSED/
                133165  011  040  120
                133170  103  040  011
                133173  011  101  103
                133176  103  105  123
                133201  123  105  104
                133204  000
     21185      133205  040  124  105  DH134:  .ASCII  / TEST  ERROR    # OF                DATA  PCR   ADDRESS/<15><12>
                133210  123  124  011
                133213  105  122  122
                133216  117  122  011
                133221  040  043  040
                133224  117  106  011
                133227  011  040  040
                133232  040  104  101
                133235  124  101  040
                133240  040  120  103
                133243  122  040  040
                133246  040  101  104
                133251  104  122  105
                133254  123  123  015
                133257  012
     21186      133260  040  040  043          .ASCIZ  /  #     PC     ERRORS  PATTERN    READ        ACCESSED/
                133263  011  040  120
                133266  103  040  011
                133271  105  122  122
                133274  117  122  123
                133277  040  040  120
                133302  101  124  124
                133305  105  122  116
                133310  040  040  040
                133313  040  122  105
                133316  101  104  040
                133321  040  040  040
                133324  040  040  040
                133327  040  101  103
                133332  103  105  123
                133335  123  105  104
                133340  000
     21187
     21188                               .EVEN
     21189      133342  001162  001116  000000  DT1:    .WORD   $TMP1,$ERRPC,0
```

GLOBAL ERROR MESSAGES

```
21190 133350  001162  001116  000001  DT4:    .WORD   $TMP1,$ERRPC,R1,CCR,0
      133356  177746  000000
21191 133362  001162  001116  000002  DT5:    .WORD   $TMP1,$ERRPC,R2,R1,$GDDAT,0
      133370  000001  001124  000000
21192 133376  001162  001116  001122  DT7:    .WORD   $TMP1,$ERRPC,$BDADR,MSER,0
      133404  177744  000000
21193 133410  001162  001116  001124  DT14:   .WORD   $TMP1,$ERRPC,$GDDAT,TSTLOC,0
      133416  003162  000000
21194 133422  001162  001116  001124  DT17:   .WORD   $TMP1,$ERRPC,$GDDAT,RECDAT,0
      133430  111464  000000
21195 133434  001162  001116  000003  DT24:   .WORD   $TMP1,$ERRPC,R3,0
      133442  000000
21196 133444  001162  001116  177744  DT27:   .WORD   $TMP1,$ERRPC,MSER,R3,0
      133452  000003  000000
21197 133456  001162  001116  001124  DT35:   .WORD   $TMP1,$ERRPC,$GDDAT,MSER,0
      133464  177744  000000
21198 133470  001162  001116  001126  DT41:   .WORD   $TMP1,$ERRPC,$BDDAT,0
      133476  000000
21199 133500  001162  001116  000001  DT43:   .WORD   $TMP1,$ERRPC,R1,RECDAT,$BDADR,0
      133506  111464  001122  000000
21200 133514  001162  001116  172354  DT47:   .WORD   $TMP1,$ERRPC,KIPAR6,$BDADR,0
      133522  001122  000000
21201 133526  001162  001116  000001  DT50:   .WORD   $TMP1,$ERRPC,R1,$BDADR,0
      133534  001122  000000
21202 133540  001162  001116  001124  DT51:   .WORD   $TMP1,$ERRPC,$GDDAT,PCR,0
      133546  177522  000000
21203 133552  001162  001116  001124  DT52:   .WORD   $TMP1,$ERRPC,$GDDAT,BCSR,0
      133560  177520  000000
21204 133564  001162  001116  001124  DT64:   .WORD   $TMP1,$ERRPC,$GDDAT,LKSFL,0
      133572  002722  000000
21205 133576  001162  001116  001124  DT65:   .WORD   $TMP1,$ERRPC,$GDDAT,0
      133604  000000
21206 133606  001162  001116  001124  DT75:   .WORD   $TMP1,$ERRPC,$GDDAT,$BDDAT,0
      133614  001126  000000
21207 133620  001162  001116  177562  DT105:  .WORD   $TMP1,$ERRPC,RBUF,0
      133626  000000
21208 133630  001162  001116  001126  DT115:  .WORD   $TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR,0
      133636  172354  001122  000000
21209 133644  001162  001122  000000  DT130:  .WORD   $TMP1,$BDADR,0
21210 133652  001162  001116  003022  DT134:  .WORD   $TMP1,$ERRPC,ERRCNT,R3,$BDDAT,R4,$BDADR,0
      133660  000003  001126  000004
      133666  001122  000000
```

MODIFIED ERROR MESSAGE TYPEOUT ROUTINE

```
                                        .SBTTL  MODIFIED ERROR MESSAGE TYPEOUT ROUTINE
                                        ;;**********************************************************************
                                        ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                        ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                        ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
                                        ;*
                                        ;*THE ONLY DIFFERENCE BETWEEN THIS ROUTINE AND THE ORIGINAL "$ERRTYP" FROM
                                        ;*SYSMAC IS THAT YOU CAN PASS INFORMATION IN GENERAL PURPOSE REGISTERS TO THIS
                                        ;*ROUTINE. THE GENERAL PURPOSE REGISTERS USED ARE TO BE SPECIFIED IN DT*
                                        ;*FORMAT. R0 SHOULD NOT BE USED.

133672                                  ERTYPE:
133672  005037  001162                         CLR     $TMP1           ;;JUST CLEAR IT
133676  113737  001102  001162                 MOVB    $TSTNM,$TMP1    ;;STORE TEST NUMBER
133704  104401  001175                         TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
133710  010046                                 MOV     R0,-(SP)        ;;SAVE R0
133712  005000                                 CLR     R0              ;;PICKUP THE ITEM INDEX
133714  153700  001114                         BISB    @#$ITEMB,R0
133720  001004                                 BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                       ;;TYPE THE PC OF THE ERROR
133722  013746  001116                         MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                       ;;ERROR ADDRESS
133726  104402                                 TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
133730  000426                                 BR      6$              ;;GET OUT
133732  005300                          1$:    DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
133734  006300                                 ASL     R0              ;;      WORK FOR THE ERROR TABLE
133736  006300                                 ASL     R0
133740  006300                                 ASL     R0
133742  062700  001324                         ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
133746  012037  133756                         MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
133752  001404                                 BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
133754  104401                                 TYPE
133756  000000                          2$:    .WORD   0               ;;ERROR MESSAGE POINTER GOES HERE
133760  104401  001175                         TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
133764  012037  133774                  3$:    MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
133770  001404                                 BEQ     5$              ;;SKIP TYPEOUT IF 0
133772  104401                                 TYPE                    ;;TYPE THE "DATA HEADER"
133774  000000                          4$:    .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
133776  104401  001175                         TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
134002  011000                          5$:    MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
134004  001004                                 BNE     7$              ;;GO TYPE THE DATA
134006  012600                          6$:    MOV     (SP)+,R0        ;;RESTORE R0
134010  104401  001175                         TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
134014  000207                                 RTS     PC              ;;RETURN
134016                                  7$:
134016  021027  000005                         CMP     (R0),#5         ;;GENERAL PURPOSE REGISTER?
134022  101021                                 BHI     9$              ;;IF NOT, GO TYPE DATA
134024  042737  000700  134060                 BIC     #BIT8!BIT7!BIT6,8$      ;;CLEAR BITS FOR SOURCE REGISTER
134032  011037  001160                         MOV     (R0),$TMP0      ;;SAVE (R0)
134036  000337  001160                         SWAB    $TMP0           ;;GET REGISTER NUMBER TO HIGH BYTE
134042  006237  001160                         ASR     $TMP0           ;;GET REGISTER NUMBER TO BITS 8-6
134046  006237  001160                         ASR     $TMP0
134052  053737  001160  134060                 BIS     $TMP0,8$        ;;SET BITS IN MOV INSTRUCTION
                                                                       ;;ACCORDING TO REGISTER NUMBER
134060  010046                          8$:    MOV     R0,-(SP)        ;;MOVE CONTEXT OF REGISTER TO STACK
134062  005720                                 TST     (R0)+           ;;ADVANCE POINTER
134064  000401                                 BR      10$             ;;GO TYPE
```

K1

MODIFIED ERROR MESSAGE TYPEOUT ROUTINE

```
21269 134066  013046                  9$:     MOV     @(R0)+,-(SP)    ;;IF NOT GPR, SAVE @(R0)+ FOR TYPEOUT
21270 134070  104402                  10$:    TYPOC                   ;;GO TYPE--OCTAL ASCI (ALL DIGITS)
21271 134072  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
21272 134074  001744                          BEQ     6$              ;;BR IF NO
21273 134076  104401  134104                  TYPE    ,11$            ;;TYPE TWO(2) SPACES
21274 134102  000745                          BR      7$
21275 134104     040     040     000  11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
21276                                          .EVEN
21277
21278                                 .SBTTL  GLOBAL SUBROUTINES SECTION
21279
21280                                 ;++
21281                                 ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
21282                                 ; THAT ARE USED IN MORE THAN ONE TEST.
21283                                 ;--
21284
21285                                 ;++
21286                                 ; FUNCTIONAL DESCRIPTION:
21287                                 ;    SUBROUTINE TO INITIALIZE ALL THE MMU REGISTERS
21288
21289
21290                                 ; INPUTS: NONE
21291
21292                                 ; OUTPUTS: NONE
21293
21294                                 ; SUBORDINATE ROUTINES USED: LOAD PARS
21295                                 ;                            LOAD PDRS
21296
21297                                 ; FUNCTIONAL SIDE EFFECTS: NONE
21298
21299                                 ; CALLING SEQUENCE:     JSR     PC,INITMM
21300
21301 134110  012701  172240  INITMM: MOV     #172240,R1              ;BASE ADDRESS OF SIPARS
21302 134114  004737  134252          JSR     PC,     LDPARS
21303 134120  012701  172260          MOV     #172260,R1              ;BASE ADDRESS OF SDPARS
21304 134124  004737  134252          JSR     PC,     LDPARS
21305 134130  012701  172340          MOV     #172340,R1              ;BASE ADDRESS OF KIPARS
21306 134134  004737  134252          JSR     PC,     LDPARS
21307 134140  012701  172360          MOV     #172360,R1              ;BASE ADDRESS OF KDPARS
21308 134144  004737  134252          JSR     PC,     LDPARS
21309 134150  012701  177640          MOV     #177640,R1              ;BASE ADDRESS OF UIPARS
21310 134154  004737  134252          JSR     PC,     LDPARS
21311 134160  012701  177660          MOV     #177660,R1              ;BASE ADDRESS OF UDPARS
21312 134164  004737  134252          JSR     PC,     LDPARS
21313 134170  012701  177600          MOV     #177600,R1              ;BASE ADDRESS OF UIPDRS
21314 134174  004737  134302          JSR     PC,     LDPDRS
21315 134200  012701  177620          MOV     #177620,R1              ;BASE ADDRESS OF UDPDRS
21316 134204  004737  134302          JSR     PC,     LDPDRS
21317 134210  012701  172300          MOV     #172300,R1              ;BASE ADDRESS OF KIPDRS
21318 134214  004737  134302          JSR     PC,     LDPDRS
21319 134220  012701  172320          MOV     #172320,R1              ;BASE ADDRESS OF KDPDRS
21320 134224  004737  134302          JSR     PC,     LDPDRS
21321 134230  012701  172200          MOV     #172200,R1              ;BASE ADDRESS OF SIPDRS
21322 134234  004737  134302          JSR     PC,     LDPDRS
21323 134240  012701  172220          MOV     #172220,R1              ;BASE ADDRESS OF SDPDRS
21324 134244  004737  134302          JSR     PC,     LDPDRS
21325 134250  000207                  RTS     PC                      ;RETURN
```

```
21327                                   ;++
21328                                   ; FUNCTIONAL DESCRIPTION:
21329                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE ADDRESS REGISTERS (PARS).
21330                                   ;       THIS ROUTINE WILL INITIALIZE 8 PARS STARTING AT A BASE ADDRESS
21331                                   ;       SUPPLIED BY THE CALLING ROUTINE. PARS 0-5 WILL BE MAPPED FROM
21332                                   ;       ADDRESS 0 TO ADDRESS 137777 (0-24K). PAR 6 WILL BE MAPPED FROM
21333                                   ;       ADDRESS 200000 TO 217777 AND PAR 7 WILL BE MAPPED TO THE I/O
21334                                   ;       PAGE.
21335                                   ;
21336                                   ; INPUTS:
21337                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PARS TO BE INITIALIZED
21338                                   ;
21339                                   ; OUTPUTS: NONE
21340                                   ;
21341                                   ; SUBORDINATE ROUTINES USED: NONE
21342                                   ;
21343                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21344                                   ;
21345                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21346                                   ;
21347 134252  012702  000006           LDPARS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21348 134256  005003                           CLR     R3                      ;INITIALIZE INDEX VALUE
21349 134260  010321                   1$:     MOV     R3,     (R1)+           ;LOAD PARS
21350 134262  062703  000200                   ADD     #200,   R3              ;INDEX IN 4K INCREMENTS
21351 134266  077204                           SOB     R2,     1$              ;LOAD FIRST SIX PARS
21352 134270  012721  002000                   MOV     #2000,  (R1)+           ;LET PAR6 MAP TO 200000
21353 134274  012711  177600                   MOV     #177600,(R1)            ;LET PAR7 MAP TO I/O PAGE
21354 134300  000207                           RTS     PC                      ;RETURN
```

GLOBAL SUBROUTINES SECTION

```
21356                                   ;++
21357                                   ; FUNCTIONAL DESCRIPTION:
21358                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE DECRIPTOR REGISTERS (PDRS).
21359                                   ;       THIS ROUTINE WILL INITIALIZE 8 PDRS STARTING AT A BASE ADDRESS
21360                                   ;       SUPPLIED BY THE CALLING ROUTINE. PDRS 0-5 WILL BE INITIALIZED TO
21361                                   ;       4K READ/WRITE BYPASS AND PDRS 6 AND 7 WILL BE INITIALIZED TO
21362                                   ;       4K READ/WRITE NO BYPASS.
21363                                   ;       NOTE:   THERE IS NO NEED TO BYPASS ON I/O PAGE REFERENCES BECAUSE
21364                                   ;               THE CACHE DOES NOT ALLOCATE ANY OF THESE REFERENCES.
21365
21366                                   ; INPUTS:
21367                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PDRS TO BE INITIALIZED
21368
21369                                   ; OUTPUTS: NONE
21370
21371                                   ; SUBORDINATE ROUTINES USED: NONE
21372
21373                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21374
21375                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21376
21377 134302  012702  000006          LDPDRS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21378 134306  012721  177406          1$:     MOV     #177406,(R1)+           ;LOAD PDRS WITH 4K READ/WRITE BYPASS
21379 134312  077203                          SOB     R2,     1$              ;LOAD FIRST SIX PDRS
21380 134314  012721  077406                  MOV     #77406, (R1)+           ;LET PAR6 BE 4K READ/WRITE NO BYPASS
21381 134320  012711  077406                  MOV     #77406, (R1)            ;LET PAR7 BE 4K READ/WRITE NO BYPASS ALSO
21382 134324  000207                          RTS     PC                      ;RETURN
```

GLOBAL SUBROUTINES SECTION

```
21384                                   ;++
21385                                   ; FUNCTIONAL DESCRIPTION:
21386                                   ;    SUBROUTINE TO HANDLE PARITY ERROR ABORTS FROM THE RAM STORE RAM TESTS.
21387                                   ;
21388                                   ; INPUTS:
21389                                   ;       MEMORY SYSTEM ERROR REGISTER CONTAINS BITS INDICATING FAILURE
21390                                   ;
21391                                   ; OUTPUTS: NONE
21392                                   ;
21393                                   ; SUBORDINATE ROUTINES USED: NONE
21394                                   ;
21395                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21396                                   ;
21397                                   ; CALLING SEQUENCE: CALLED BY PARITY ABORT
21398                                   ;       MOV     @#114, SLOC00  ;SAVE CONTENTS OF PARITY ABORT VECTOR
21399                                   ;       MOV     #DSPAR, @#114  ;LET VECTOR POINT TO PARITY ABORT ROUTINE
21400                                   ;
21401                                   ;       (CACHE PARITY ERROR OCCURS)
21402
21403 134326  011637  001122           RAMPAR: MOV     (SP),$BDADR             ;STOR ADDESS TRAPPED
21404 134332  032737  000100  177744           BIT     #BIT06, MSER            ;IF LOW BYTE PARITY ERROR
21405 134340  001401                           BEQ     1$                      ;THEN
21406 134342  104007                           ERROR   +7                      ;ERROR
21407 134344  032737  000200  177744   1$:     BIT     #BIT07, MSER            ;IF HIGH BYTE PARITY ERROR
21408 134352  001401                           BEQ     2$                      ;THEN
21409 134354  104007                           ERROR   +7                      ;ERROR
21410 134356  032737  000040  177744   2$:     BIT     #BIT05, MSER            ;IF TAG PARITY ERROR
21411 134364  001401                           BEQ     3$                      ;THEN
21412 134366  104007                           ERROR   +7                      ;ERROR
21413 134370  005037  177744           3$:     CLR     MSER                    ;INITIALIZE MSER AFTER ERROR
21414 134374  000002                           RTI                             ;RETURN
21415 134376  005237  002722           LKSINT: INC     LKSFL                   ;INCREMENT FLAG
21416 134402  000002                           RTI
21417
21418                                   .SBTTL Q22BE SIZE ROUTINE
21419                                   ;THIS ROUTINE WILL AUTOSIZE FOR UP TO TWO Q22 BUS EXERCISERS.  IF NONE
21420                                   ;FOUND LOCATIONS CSR1 AND CSR12 WILL BE LEFT ZEROES.  THIS ROUTINE WILL
21421                                   ;ONLY RUN IN NOT UFD MODE.
21422
21423 134404  032737  001000  177750   Q22SIZ: BIT     #BIT09,MAIREG                   ;UNIBUS SYSTEM?
21424 134412  001401                           BEQ     1$                              ;IF NOT, ADVANCE TO ROUTINE
21425 134414  000207                           RTS     PC                              ;OTHERWISE, RETURN
21426                                   ;
21427                                   ; PREPARE TO DO SIZING
21428                                   ;
21429 134416  013701  000004           1$:     MOV     ERRVEC,R1               ;STORE TIMEOUT VECTOR
21430 134422  012737  134576  000004           MOV     #7$,ERRVEC              ;POINT NEW TO PROGRAM
21431 134430  012737  000340  000006           MOV     #340,ERRVEC+2           ;AT PRIORITY 7
21432 134436  005037  001160                   CLR     $TMP0                   ;CLEAR Q22BE COUNTER
21433 134442  012702  170000                   MOV     #170000,R2              ;FIRST POSSIBLE ADDRESS
21434 134446  012703  000510                   MOV     #510,R3                 ;VECTOR FOR IT
21435 134452  000404                           BR      3$                      ;TRY THOSE VALUES
21436                                   ;
21437                                   ; NOW DO ACTUAL SIZING
21438                                   ;
21439 134454  062702  000020           2$:     ADD     #20,R2                  ;GET CSR FOR NEXT Q22BE
21440 134460  062703  000004                   ADD     #4,R3                   ;GET VECTOR FOR NEXT ONE
```

Q22BE SIZE ROUTINE

```
21441 134464  005712                    3$:     TST     (R2)                    ;TRY TO ACCESS CSR
21442                                   ;
21443                                   ; IF NO TIMEOUT, STORE EXISTING ADDRESSES TO REGISTERS
21444                                   ;
21445 134466  005737  001160                    TST     $TMP0                   ;FIRST Q22BE FOUND?
21446 134472  001010                            BNE     4$                      ;IF SECOND, BRANCH
21447 134474  012705  002664                    MOV     #CSR1,R5                ;START WITH CSR1 FOR 1ST
21448 134500  010237  002702                    MOV     R2,SIMGOA               ;SIMULTANEOUS GO
21449 134504  062737  000016  002702            ADD     #16.,SIMGOA             ;ADDRESS
21450 134512  000402                            BR      5$                      ;BRANCH TO INITIALISE
21451 134514  012705  002704            4$:     MOV     #CSR12,R5               ;START WITH CSR12 FOR 2ND
21452 134520  012704  000004            5$:     MOV     #4,R4                   ;INITIALISE 5 REGISTERS
21453 134524  010215                            MOV     R2,(R5)                 ;INITIALISE CSR1
21454 134526  011565  000002            6$:     MOV     (R5),2(R5)              ;STORE TO NEXT ONE
21455 134532  005725                            TST     (R5)+                   ;GET NEXT ADDRESS
21456 134534  062715  000002                    ADD     #2,(R5)                 ;GET ADDRESS, POINT NEXT
21457 134540  077406                            SOB     R4,6$                   ;DO FOR NEXT 4 REGISTERS
21458 134542  010365  000002                    MOV     R3,2(R5)                ;STORE INTERRUPT VECTOR
21459 134546  010365  000004                    MOV     R3,4(R5)                ;AND PRIORITY
21460 134552  062765  000002  000004            ADD     #2,4(R5)
21461 134560  005237  001160                    INC     $TMP0                   ;COUNT Q22BE'S
21462 134564  022737  000002  001160            CMP     #2,$TMP0                ;TWO FOUND?
21463 134572  001406                            BEQ     9$                      ;IF SO, STOP SIZING
21464 134574  000402                            BR      8$                      ;OTHERWISE, CONTINUE SIZING
21465                                   ;
21466                                   ; ON TIMEOUT TRY TO LOOK AT NEXT ADDRESS RANGE
21467                                   ;
21468 134576  005726                    7$:     TST     (SP)+                   ;RESTORE STACK FROM
21469 134600  005726                            TST     (SP)+                   ;TIMEOUT
21470 134602  022702  170160            8$:     CMP     #170160,R2              ;AT THE LAST POSSIBLE?
21471 134606  001322                            BNE     2$                      ;IF NOT, BRANCH
21472 134610  005737  002664            9$:     TST     CSR1                    ;1 FOUND?
21473 134614  001402                            BEQ     10$                     ;IF NONE, BRANCH
21474 134616  104401  134630                    TYPE    ,ONEQ22                 ;TYPE FOUND
21475 134622  010137  000004            10$:    MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
21476 134626  000207                            RTS     PC                      ;RETURN
21477
21478 134630      012     015     121   ONEQ22: .ASCIZ  <12><15>/Q22BE USED DURING TESTING/
      134633      062     062     102
      134636      105     040     125
      134641      123     105     104
      134644      040     104     125
      134647      122     111     116
      134652      107     040     124
      134655      105     123     124
      134660      111     116     107
      134663      000
21479                                   .EVEN
21480                                   .SBTTL Q22BE INTERRUPT INITIALISE ROUTINE
21481                                   ;THIS ROUTINE WILL INITIALISE Q22BE TO INTERRUPT AT A PRIORITY AT (R0)+
21482                                   ;AT THE STARTING ADDRESS IN R3. THE TEST HAVE TO SET ACTUAL DONE BIT
21483                                   ;BY CLEARING GO.
21484
21485 134664  005013                    Q22INT: CLR     (R3)                    ;CLEAR TRANSFER TYPE IN CSR1
21486 134666  052710  000001                    BIS     #BIT00,(R0)             ;ZERO DONE
21487 134672  011063  000002                    MOV     (R0),2(R3)              ;SET PRIORITY IN CSR2
21488 134676  042710  000001                    BIC     #BIT00,(R0)             ;PREPARE TO SET DONE
```

Q22BE INTERRUPT INITIALISE ROUTINE

```
21489 134702  000207                          RTS     PC
21490
21491                                 .SBTTL DMATRN DATO CYCLE THRU Q22BE
21492                                 ;THIS ROUTINE PERFORMS DATO FROM A LOCATION TEMP THRU THE FIRST
21493                                 ;FOUND Q22BE STARTING AT LOCATION @CSR1. R0 HAS 0 IF ONLY 1 TRANSFER IS
21494                                 ;TO BE PERFORMED. OTHERWISE 16 BLOCK MODE TRANSFERS ARE TO BE PERFORMED.
21495                                 ;IN THE LATTER CASE ADDRESS AND WORD COUNT HAS TO BE LOADED BEFORE.
21496
21497 134704  012777  012525  045762  DMATRN: MOV     #12525,@DATA                    ;DATA USED
21498 134712  005700                          TST     R0                              ;DO 1 WORD?
21499 134714  001404                          BEQ     1$                              ;IF YES, BRANCH
21500 134716  012777  001001  045742          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE, GO
21501 134724  000414                          BR      2$                              ;BRANCH TO DO IT
21502 134726  012777  001601  045730  1$:     MOV     #1601,@CSR1                     ;RESET LATENCY COUNT,DATO
21503 134734  012777  002740  045726          MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21504 134742  012777  177777  045722          MOV     #177777,@WC                     ;DO 1 WORD
21505 134750  012777  000001  045710          MOV     #BIT00,@CSR2                    ;DO IT
21506 134756  105777  045704          2$:     TSTB    @CSR2                           ;DMA DONE?
21507 134762  100375                          BPL     2$                              ;WAIT TILL DONE
21508 134764  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21509
21510                                 .SBTTL DMARD DATI THRU Q22BE
21511                                 ;THIS ROUTINE PERFORMS DATI CYCLE THRU Q22BE IN EITHER BLOCK MODE OR A SINGLE
21512                                 ;TRANSFER MODE. MEMORY LOCATION USED IS TEMP. R0 IS ZERO FOR SINGLE TRANSFER
21513
21514 134766  012777  002740  045674  DMARD:  MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21515 134774  005700                          TST     R0                              ;DO 1 WORD?
21516 134776  001412                          BEQ     1$                              ;IF YES, BRANCH
21517 135000  012777  001507  045656          MOV     #1507,@CSR1                     ;16 DATIB
21518 135006  012777  177770  045656          MOV     #177770,@WC                     ;DO 8 WORD
21519 135014  012777  001001  045644          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE GO
21520 135022  000411                          BR      2$                              ;GO CHECK
21521 135024  012777  001407  045632  1$:     MOV     #1407,@CSR1                     ;RESET LATENCY COUNT,DATI
21522                                                                                 ;LOAD NEW DATA TO DATA R.
21523 135032  012777  177777  045632          MOV     #177777,@WC                     ;DO 1 WORD
21524 135040  012777  000001  045620          MOV     #BIT00,@CSR2                    ;DO IT
21525 135046  105777  045614          2$:     TSTB    @CSR2                           ;DMA DONE?
21526 135052  100375                          BPL     2$                              ;WAIT TILL DONE
21527 135054  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21528
21529
21530
21531 135056  011637  001122          TOUT:   MOV     (SP),@BDADR             ;STORE TRAPPED PC
21532 135062  104130                          ERROR   .130                    ;UNEXPECTED TRAP
21533 135064  000002                          RTI
21534
21536                                 ;
21537                                 ;MMU GLOBAL SUBROUTINES
21538                                 ;
21539                                 ;
21540                                 ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT
21541                                 ;
21542 135066  010046          MMU:    MOV     R0,-(SP)                ;SAVE CONTENTS OF REGISTERS
21543 135070  010146                  MOV     R1,-(SP)                ;
21544 135072  010246                  MOV     R2,-(SP)                ;
21545 135074  012700  177600          MOV     #177600,R0              ;
21546 135100  004737  135166          JSR     PC,PDR                  ;INIT I AND D USER PDR'S
```

DMARD DATI THRU Q22BE

```
21547 135104  004737  135210                    JSR     PC,PAR          ;INIT I USER PAR'S
21548 135110  004737  135210                    JSR     PC,PAR          ;INIT D USER PAR'S
21549 135114  012700  172200                    MOV     #172200,R0
21550 135120  004737  135166                    JSR     PC,PDR          ;INIT I AND D SUP PDR'S
21551 135124  004737  135210                    JSR     PC,PAR          ;INIT I SUP PAR'S
21552 135130  004737  135210                    JSR     PC,PAR          ;INIT D SUP PAR'S
21553 135134  004737  135166                    JSR     PC,PDR          ;INIT I AND D KER PDR'S
21554 135140  004737  135210                    JSR     PC,PAR          ;INIT I KER PAR'S
21555 135144  004737  135210                    JSR     PC,PAR          ;INIT D KER PAR'S
21556 135150  012737  000027  172516            MOV     #27,@#172516    ;INIT MMR3
21557 135156  012602                            MOV     (SP)+,R2        ;RESTORE REGISTERS
21558 135160  012601                            MOV     (SP)+,R1        ;
21559 135162  012600                            MOV     (SP)+,R0        ;
21560 135164  000207                            RTS     PC              ;RETURN
21561                                 ;
21562                                 ;ROUTINE TO INITIALIZE PDR'S
21563                                 ;
21564 135166  005002                  PDR:      CLR     R2              ;INIT CNTR
21565 135170  012720  077406          PDR1:     MOV     #77406,(R0)+    ;INIT PDR
21566 135174  062702  000001                    ADD     #1,R2           ;INCREMENT CNTR
21567 135200  022702  000020                    CMP     #16.,R2         ;ARE WE DONE?
21568 135204  001371                            BNE     PDR1            ;BRANCH IF NOT
21569 135206  000207                            RTS     PC              ;RETURN
21570                                 ;
21571                                 ;ROUTINE TO INITIALIZE PAR'S
21572                                 ;
21573 135210  005001                  PAR:      CLR     R1              ;SETUP TO INIT PAR
21574 135212  010120                  PAR1:     MOV     R1,(R0)+        ;INIT PAR
21575 135214  062701  000200                    ADD     #200,R1         ;GET READY FOR NEXT PAR
21576 135220  022701  001600                    CMP     #1600,R1        ;REACHED A PAR7
21577 135224  001372                            BNE     PAR1            ;BRANCH IF NOT
21578 135226  012720  177600                    MOV     #177600,(R0)+   ;INIT PAR7
21579 135232  000207                            RTS     PC              ;RETURN
21580                                 ;
21581                                 ;TIME OUT ROUTINE
21582                                 ;
21583 135234  005205                  ADDTRP:   INC     R5              ;INCREMENT TIME OUT FLAG
21584 135236  000002                            RTI                     ;RETURN
21585                                 ;
21586                                 ;MMU TRAP ROUTINE
21587                                 ;
21588 135240  023727  003030  000001  MMUTRP:   CMP     FLAG,#1         ;ARE WE EXPECTING AN ABORT
21589 135246  001401                            BEQ     1$              ;YES GO ON
21590 135250  104002                            ERROR   +2              ;NO GO TO ERROR
21591 135252  010046                  1$:       MOV     R0,-(SP)        ;SAVE CONTENTS OF REG 0
21592 135254  013700  177776                    MOV     @#177776,R0     ;SAVE A COPY OF PSW
21593 135260  072027  177764                    ASH     #-14,R0         ;LOOK AT BITS<15:14>
21594 135264  020027  000002                    CMP     R0,#2           ;WAS PS<15:14>=10
21595 135270  001001                            BNE     OK              ;NO GO ON
21596 135272  000411                            BR      NOTOK           ;YES CHANGE BITS TO 00
21597 135274  013700  177776          OK:       MOV     @#177776,R0     ;SAVE A COPY OF PSW
21598 135300  072027  000002                    ASH     #2,R0           ;LOOK AT BITS<13:12>
21599 135304  072027  177764                    ASH     #-14,R0         ;
21600 135310  020027  000002                    CMP     R0,#2           ;WAS PS<13:12>=10
21601 135314  001002                            BNE     OK1             ;NO GO ON
21602 135316  005066  000004          NOTOK:    CLR     4(SP)           ;CLEAR ILLEGAL MODE FFROM OLD PSW
21603 135322  013737  177572  003042  OK1:      MOV     @#177572,SAVMR0 ;SAVE A COPY OF MMR0
```

DMARD DATI THRU Q228E

```
21604 135330  013737  177574  003044          MOV     @#177574,SAVMR1         ;SAVE A COPY OF MMR1
21605 135336  013737  177576  003046          MOV     @#177576,SAVMR2         ;SAVE A COPY OF MMR2
21606 135344  005037  177572                  CLR     @#177572                ;CLEAR ABORT BITS AND TURN MMU OFF
21607 135350  005037  003030                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
21608 135354  012600                          MOV     (SP)+,R0                ;RESTORE ORIGINAL CONTENTS OF REG 0
21609 135356  000002                          RTI                             ;RETURN
```

F2

DMARD DATI THRU Q228E

```
21613                                   ;FPP COMMON SUBROUTINES
21614 135360  012600                    WLDTRP: MOV     (SP)+,R0        ;SAVE PC
21615 135362  012605                            MOV     (SP)+,R5        ;SAVE STATUS AND RESTORE STACK
21616 135364  104003                            ERROR   +3
21617 135366  000110                            JMP     (R0)            ;GO BACK INLINE
21618                                   ;
21619                                   ;
21620                                   ;
21621 135370  000000                    TRPFLG: .WORD   0
21622 135372  000207                    ERRFP:  RTS     R7
21623 135374  000207                    ERR:    RTS     R7
21624                                   ;
21625                                   ;
21626                                   ;
21627                                   ;
21628                                   ;
21629                                   ;SUBROUTINE DATA VERFICATION -
21630                                   ;
21631                                   ; CALLED BY     JSR     R7,DATVER
21632                                   ;
21633                                   ;INPUT:         (R4)=EXPECTED DATA
21634                                   ;               (R1)=RECEIVED DATA
21635                                   ;
21636                                   ;THIS ROUTINE VERIFIES THAT THE 4 CONSECTIVE WORDS STARTING WITH (R4) ARE
21637                                   ;EQUAL TO THE FOUR WORDS ADDRESSED BY (R1). THE CONTENTS OF R4, AND R1 ARE NOT
21638                                   ;DISTURBED.
21639                                   ;LOCATION "COUNT" , IF NOT EQUAL TO 0 SIGNIFIES DATA ERROR
21640                                   ;IF THE STATUS IS FLOATING MODE, THE LAST TWO BYTES OF RECEIEVED
21641                                   ;ARE SIMPLY CHECKED FOR ZEROS
21642                                   ;
21643
21644 135376  010446                    DATVFR: MOV     R4,-(SP)        ;SAVE R4
21645 135400  010146                            MOV     R1,-(SP)                ;SAVE R1
21646 135402  012737  000003  003120            MOV     #3,COUNT        ;SET UP ITERATION COUNT
21647 135410  000137  135426                    JMP     DAT1            ;
21648                                   ;
21649 135414  010446                    DATVER: MOV     R4,-(SP)        ;SAVE R4
21650 135416  010146                            MOV     R1,-(SP)        ;SAVE R1
21651 135420  012737  000005  003120            MOV     #5,COUNT        ;SET UP ITERATION COUNT
21652 135426  005337  003120            DAT1:   DEC     COUNT
21653 135432  001402                            BEQ     2$              ;BRANCH IF DONE
21654 135434  022421                            CMP     (R4)+,(R1)+             ;
21655 135436  001773                            BEQ     DAT1
21656 135440  012601                    2$:     MOV     (SP)+,R1        ;RESTORE R1
21657 135442  012604                            MOV     (SP)+,R4        ;RESTORE R4
21658 135444  000207                            RTS     R7              ;GO BACK TO CALLING ROUTINE
21659                                                                   ;IF DATA ERROR, COUNT NE 0
```

```
21662
21663
21664                                   .SBTTL  END OF PASS ROUTINE
                                        ;;*****************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO LOOP
135446                                  $EOP:
135446  032737  000100  000052                 BIT     #BIT06,@#52
135454  001030                                  BNE     $GET42
135456  004537  124272                         JSR     R5,VIREOP
135462  005037  001102                         CLR     $TSTNM          ;;ZERO THE TEST NUMBER
135466  005037  001164                         CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
135472  005237  001206                         INC     $PASS           ;;INCREMENT THE PASS NUMBER
135476  042737  100000  001206                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
135504  005327                                  DEC     (PC)+           ;;LOOP?
135506  000001                          $EOPCT: .WORD   1
135510  003022                                  BGT     $DOAGN          ;;YES
135512  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
135514  000001                          $ENDCT: .WORD   1
135516  135506                                  $EOPCT
135520  104401  135565                         TYPE    ,$ENDMG         ;;TYPE "END PASS #"
135524  013746  001206                         MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
135530  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
135532  104401  135562                         TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
135536  013700  000042                 $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
135542  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
135544  000005                                  RESET                   ;;CLEAR THE WORLD
135546  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
135550  000240                                  NOP                     ;;SAVE ROOM
135552  000240                                  NOP                     ;;FOR
135554  000240                                  NOP                     ;;ACT11
135556                                  $DOAGN:
135556  000137                                  JMP     @(PC)+          ;;RETURN
135560  004724                          $RTNAD: .WORD   LOOP
135562     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
135565     015     012     105          $ENDMG: .ASCIZ  <15><12>/END PASS #/
135570     116     104     040
135573     120     101     123
135576     123     040     043
135601     000
21665                                   .SBTTL  SCOPE HANDLER ROUTINE
                                        ;;*****************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*SW08=1         LOOP ON TEST IN SWR<5:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
135602                                  $SCOPE:
135602  104407                                  CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
```

SCOPE HANDLER ROUTINE

```
135604  052737  001000  177520          BIS #1000,BCSR ;ENABLE
135612  032777  040000  043320  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
135620  001117                          BNE     $OVER           ;;YES IF SW14=1
                                ;*****START OF CODE FOR THE XOR TESTER*****
135622  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
135624  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
135630  012737  135650  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
135636  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
135642  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
135646  000466                          BR      $SVLAD          ;;GO TO THE NEXT TEST
135650  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
135652  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
135656  000426                          BR      7$              ;;LOOP ON THE PRESENT TEST
135660                          6$:;*****END OF CODE FOR THE XOR TESTER*****
135660  032777  000400  043252          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
135666  001407                          BEQ     2$              ;;BR IF NO
135670  017746  043244                  MOV     @SWR,-(SP)      ;;SET DESIRED TEST NUM. FROM SWR
135674  042716  000300                  BIC     #$SWRMK,(SP)    ;;STRIP AWAY UNDESIRED BITS
135700  122637  001102                  CMPB    (SP)+,$TSTNM    ;;ON THE RIGHT TEST?
135704  001465                          BEQ     $OVER           ;;BR IF YES
135706  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
135712  001421                          BEQ     3$              ;;BR IF NO
135714  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
135722  101015                          BHI     3$              ;;BR IF NO
135724  032777  001000  043206          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
135732  001404                          BEQ     4$              ;;BR IF NO
135734  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
135742  000446                          BR      $OVER
135744  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
135750  005037  001164                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
135754  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
135756  032777  004000  043154  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
135764  001011                          BNE     1$              ;;BR IF YES
135766  005737  001206                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
135772  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
135774  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
136000  023737  001164  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
136006  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
136010  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
136016  013737  136102  001164          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
136024  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
136030  113737  001102  001204          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
136036  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
136042  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
136046  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
136052  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
136060  013777  001102  043054  $OVER:  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
136066  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
136072  042737  001000  177520          BIC #1000,BCSR ;DISABLE
136100  000002                          RTI
136102  000001                  $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS
21666                           .SBTTL  ERROR HANDLER ROUTINE
                                ;;*********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO ERTYPE ON ERROR
```

ERROR HANDLER ROUTINE

```
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1        HALT ON ERROR
                                ;*SW13=1        INHIBIT ERROR TYPEOUTS
                                ;*SW10=1        BELL ON ERROR
                                ;*SW09=1        LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   +N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
136104
136104  005737  004120          $ERROR: TST     UQUIET          ;;TEST FOR USER-QUIET MODE
136110  001403                          BEQ     9$              ;;BRANCH IF FIELD-SERVICE MODE
136112  005000                          CLR     R0              ;;IN CASE R0 HAS A #3 IN IT (^C)
136114  004737  136326                  JSR     PC,ABORT        ;;TEST FOR ABORT CONDITION
136120                          9$:
136120  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
136122  052737  001000  177520          BIS     #1000,@CSR              ;ENABLE HALT ON BREAK
136130  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
136134  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
136136  013777  001102  042776          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
136144  032777  002000  042766          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
136152  001402                          BEQ     1$              ;;NO - SKIP
136154  104401  001170                  TYPE    .$BELL          ;;RING BELL
136160  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
136164  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
136170  162737  000002  001116          SUB     #2,$ERRPC
136176  117737  042714  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
136204  032777  020000  042726          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
136212  001004                          BNE     20$             ;;SKIP TYPEOUTS
136214  004737  133672                  JSR     PC,ERTYPE       ;;GO TO USER ERROR ROUTINE
136220  104401  001175                  TYPE    .$CRLF
136224                          20$:
136224  122737  000001  001220          CMPB    @APTENV,$ENV    ;;RUNNING IN APT MODE
136232  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
136234  113737  001114  136246          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
136242  004737  136504                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
136246     000          21$:    .BYTE   0
136247     000                  .BYTE   0
136250  000777          22$:    BR      22$             ;;APT ERROR LOOP
136252  005777  042662          2$:     TST     @SWR            ;;HALT ON ERROR
136256  100002                          BPL     3$              ;;SKIP IF CONTINUE
136260  000000                          HALT                    ;;HALT ON ERROR!
136262  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
136264  032777  001000  042646  3$:     BIT     @BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
136272  001402                          BEQ     4$              ;;BR IF NO
136274  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
136300  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
136304  001402                          BEQ     5$              ;;BR IF NONE
136306  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
136312                          5$:
136312  022737  135546  000042          CMP     @$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
136320  001001                          BNE     6$              ;;BRANCH IF NO
136322  000000                          HALT                    ;;YES
136324                          6$:
136324  000002                          RTI                     ;;RETURN
                                .SBTTL  ABORT ROUTINE FOR LCP/ORION UFD MODE
136326  005737  004116          ABORT:  TST     UFDFLG          ;TEST FOR USER FRIENDLY MODE
136332  001454                          BEQ     NOABRT          ;IF NOT UFD THEN CONTINUE NORMAL OPERATION
136334  020027  000032                  CMP     R0,#32          ;IS IT A ^Z ?
```

J2

ABORT ROUTINE FOR LCP/ORION UFD MODE

```
136340  001443                          BEQ     ABORTZ          ;JUST GO BACK TO CHAIN IF IT IS (NO ERROR)
136342  020027  000003                   CMP     R0,#3           ;IS IS A ↑C ?
136346  001404                          BEQ     ABORTC          ;BR TO LOAD ↑C ON XXDP+ STACK (NO ERROR)
136350  005737  004120                   TST     UQUIET          ;TEST FOR USER-QUIET MODE
136354  001443                          BEQ     NOABRT          ;IF FIELD-SERVICE MODE, CONTINUE NORMAL OPERATION
                                                                ; BECAUSE FIELD-SERVICE MODE DOES NOT QUIT ON ERROR
136356  000422                          BR      ABORTE          ;SET DRSERR THEN LEAVE
136360  013737  004112  000030  ABORTC: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
136366  013737  004114  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
136374  104043                          EMT     +43             ;GET XXDP STACK LOC. INTO R0 FROM MONITOR
136376  005720                  1$:     TST     (R0)+           ;FIND END OF STACK
136400  001376                          BNE     1$
136402  112760  000057  177777          MOVB    #'/,-1(R0)      ;LOAD SLASH OVER ZERO
136410  112720  000136                  MOVB    #'↑,(R0)+       ;LOAD UPARROW
136414  112720  000103                  MOVB    #'C,(R0)+       ;LOAD C
136420  105010                          CLRB    (R0)            ;MAKE NEW END TO STACK
136422  000412                          BR      ABORTZ          ;NOW LEAVE
136424  013737  004112  000030  ABORTE: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
136432  013737  004114  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
136440  104042                          EMT     +42             ;GET DCA LOCATION INTO R0 FROM MONITOR
136442  012760  177777  000042          MOV     #-1,42(R0)      ;SET A -1 INTO LOCATION DRSERR IN MONITOR
136450  013700  000042          ABORTZ: MOV     @#42,R0         ;AND PUT THE MONITOR RETURN ADDRESS IN R0
136454  005037  000042                  CLR     @#42            ;CLEAR MONITOR RETURN FLAG
136460  000137  135546                  JMP     $ENDAD          ;RETURN TO MONITOR-DO NOT PUSH STACK HERE
136464  000207                  NOABRT: RTS     PC              ;IF NOTUFD RETURN TO MAINLINE
21667                                   .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;**************************************************************
136466  112737  000001  136732  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
136474  112737  000001  136730  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
136502  000403                          BR      $ATYC
136504  112737  000001  136732  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
136512                          $ATYC:
136512  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
136514  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
136516  105737  136730                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
136522  001450                          BEQ     5$              ;;IF NOT:  BR
136524  122737  000001  001220          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
136532  001031                          BNE     3$              ;;IF NOT:  BR
136534  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
136542  001425                          BEQ     3$              ;;IF NOT:  BR
136544  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
136550  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
136556  005737  001200          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
136562  001375                          BNE     1$              ;;IF NOT:  WAIT
136564  010037  001214                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
136570  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
136572  001376                          BNE     2$
136574  163700  001214                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
136600  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
136602  010037  001216                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
136606  012737  000004  001200          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
136614  000413                          BR      5$
136616  017637  000004  136642  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
136624  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
136632  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
136636  004737  136734                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
136642  000000                  4$:     .WORD   0
```

K2

APT COMMUNICATIONS ROUTINE

```
136644                                          5$:
136644  105737  136732                 10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
136650  001416                                  BEQ     12$             ;;IF NOT:  BR
136652  005737  001220                          TST     $ENV            ;;RUNNING UNDER APT?
136656  001413                                  BEQ     12$             ;;IF NOT:  BR
136660  005737  001200                 11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
136664  001375                                  BNE     11$             ;;IF NOT:  WAIT
136666  017637  000004  001202                 MOV     @4(SP),$FATAL   ;;GET ERROR #
136674  062766  000002  000004                 ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
136702  005237  001200                          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
136706  105037  136732                 12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
136712  105037  136731                          CLRB    $LFLG           ;;CLEAR LOG FLAG
136716  105037  136730                          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
136722  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
136724  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
136726  000207                                  RTS     PC              ;;RETURN
136730     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
136731     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
136732     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
                                        .EVEN
             000200                     APTSIZE=200
             000001                     APTENV=001
             000100                     APTSPOOL=100
             000040                     APTCSUP=040
 21668                                  .SBTTL  TYPE ROUTINE
                                        ;;******************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*       TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*       TYPE
                                        ;*       MESADR
                                        ;*
136734  105737  001157                  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
136740  100002                                  BPL     1$              ;;BR IF YES
136742  000000                                  HALT                    ;;HALT HERE IF NO TERMINAL
136744  000430                                  BR      3$              ;;LEAVE
136746  010046                          1$:     MOV     R0,-(SP)        ;;SAVE R0
136750  017600  000002                          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
136754  122737  000001  001220                 CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
136762  001011                                  BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
136764  132737  000100  001221                 BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
136772  001405                                  BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
136774  010037  137004                          MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
137000  004737  136474                          JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
137004  000000                          61$:    .WORD   0               ;;MESSAGE ADDRESS
137006  132737  000040  001221          62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
137014  001003                                  BNE     60$             ;;YES,SKIP TYPE OUT
137016  112046                          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
137020  001005                                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
137022  005726                                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
```

TYPE ROUTINE

```
137024  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
137026  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
137032  000002                          RTI                     ;;RETURN
137034  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
137040  001430                          BEQ     8$
137042  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
137046  001006                          BNE     5$
137050  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
137052  104401                          TYPE                    ;;TYPE A CR AND LF
137054  001175                          $CRLF
137056  105037  137264                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
137062  000755                          BR      2$              ;;GET NEXT CHARACTER
137064  004737  137146            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
137070  123726  001156            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
137074  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
137076  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
137102  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
137106  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
137110  004737  137146                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
137114  105337  137264                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
137120  000770                          BR      7$              ;;LOOP
                        ;HORIZONTAL TAB PROCESSOR
137122  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
137126  004737  137146            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
137132  132737  000007  137264          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
137140  001372                          BNE     9$              ;;TAB STOP
137142  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
137144  000724                          BR      2$              ;;GET NEXT CHARACTER
137146                          $TYPEC:
137146  105777  041772                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
137152  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
137154  017746  041766                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
137160  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
137164  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
137170  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
137172                          101$:                                                           ;MJD001
137172  105777  041746                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
137176  100375                          BPL     101$                                            ;MJD001
137200  117716  041742                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
137204  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
137210  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
137214  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
137216                          102$:                                                           ;MJD001
137216  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
137220                          10$:                                                            ;MJD001
137220  105777  041724                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
137224  100375                          BPL     10$                                             ;MJD001
137226  116677  000002  041716          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
137234  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
137242  001003                          BNE     1$              ;;BRANCH IF NO
137244  105037  137264                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
137250  000406                          BR      $TYPEX          ;;EXIT
137252  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
137260  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
137262  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
137264  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
```

TYPE ROUTINE

```
        137266  000207                  $TYPEX: RTS     PC
 21669
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                        ;;**************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
        137270  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
        137274  116637  000001  137513         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
        137302  112637  137515                 MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
        137306  062716  000002                 ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
        137312  000406                         BR      $TYPON
        137314  112737  000001  137513 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
        137322  112737  000006  137515         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
        137330  112737  000005  137512 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
        137336  010346                         MOV     R3,-(SP)        ;;SAVE R3
        137340  010446                         MOV     R4,-(SP)        ;;SAVE R4
        137342  010546                         MOV     R5,-(SP)        ;;SAVE R5
        137344  113704  137515                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
        137350  005404                         NEG     R4
        137352  062704  000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
        137356  110437  137514                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
        137362  113704  137513                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
        137366  016605  000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
        137372  005003                         CLR     R3              ;;CLEAR THE OUTPUT WORD
        137374  006105                  1$:    ROL     R5              ;;ROTATE MSB INTO "C"
        137376  000404                         BR      3$              ;;GO DO MSB
        137400  006105                  2$:    ROL     R5              ;;FORM THIS DIGIT
        137402  006105                         ROL     R5
        137404  006105                         ROL     R5
        137406  010503                         MOV     R5,R3
        137410  006103                  3$:    ROL     R3              ;;GET LSB OF THIS DIGIT
        137412  105337  137514                 DECB    $OMODE          ;;TYPE THIS DIGIT?
        137416  100016                         BPL     7$              ;;BR IF NO
        137420  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
        137424  001002                         BNE     4$              ;;TEST FOR 0
        137426  005704                         TST     R4              ;;SUPPRESS THIS 0?
        137430  001403                         BEQ     5$              ;;BR IF YES
        137432  005204                  4$:    INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
        137434  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
```

BINARY TO OCTAL (ASCII) AND TYPE

```
137440  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
137444  110337  137510                  MOVB    R3,8$           ;;SAVE FOR TYPING
137450  104401  137510                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
137454  105337  137512          7$:     DECB    $OCNT           ;;COUNT BY 1
137460  003347                          BGT     2$              ;;BR IF MORE TO DO
137462  002402                          BLT     6$              ;;BR IF DONE
137464  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
137466  000744                          BR      2$              ;;GO DO THE LAST DIGIT
137470  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
137472  012604                          MOV     (SP)+,R4        ;;RESTORE R4
137474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
137476  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
137504  012616                          MOV     (SP)+,(SP)
137506  000002                          RTI                     ;;RETURN
137510     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
137511     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
137512     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
137513     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
137514  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 21670                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE
137516                          $TYPDS:
137516  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
137520  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
137522  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
137524  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
137526  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
137530  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
137534  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
137540  100004                          BPL     1$              ;;BR IF INPUT IS POS.
137542  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
137544  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
137552  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
137554  012703  137732                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
137560  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
137564  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
137566  016001  137722                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
137572  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
137574  002402                          BLT     4$              ;;BR IF DONE
137576  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
137600  000774                          BR      3$
137602  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
137604  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
137606  001002                          BNE     5$              ;;FALL THROUGH IF 0
137610  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
137612  100407                          BMI     7$              ;;BR IF YES
137614  106316                  5$:     ASLB    (SP)            ;;MSD?
137616  103003                          BCC     6$              ;;BR IF NO
137620  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
```

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```
137626  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
137632  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
137636  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
137640  005720                          TST     (R0)+           ;;JUST INCREMENTING
137642  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
137646  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
137650  003002                          BGT     8$              ;;GO TO EXIT
137652  010502                          MOV     R5,R2           ;;GET THE LSD
137654  000764                          BR      6$              ;;GO CHANGE TO ASCII
137656  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
137660  100003                          BPL     9$              ;;BR IF NO
137662  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
137670  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
137672  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
137674  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
137676  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
137700  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
137702  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
137704  104401  137732                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
137710  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
137716  012616                          MOV     (SP)+,(SP)
137720  000002                          RTI                     ;;RETURN TO USER
137722  023420                  $DTBL:  10000.
137724  001750                          1000.
137726  000144                          100.
137730  000012                          10.
137732                          $DBLK:  .BLKW   4
21671                           .SBTTL  TTY INPUT ROUTINE
                                ;;******************************************************************
                                .ENABL  LSB
                                ;;******************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
137742  022737  000176  001140  $CKSWR: CMP     @#SWREG,SWR     ;;IS THE SOFT-SWR SELECTED?
137750  001074                          BNE     15$             ;;BRANCH IF NO
137752  105777  041166                  TSTB    @#TKS           ;;CHAR THERE?
137756  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
137760  117746  041162                  MOVB    @#TKB,-(SP)     ;;SAVE THE CHAR
137764  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
137770  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
137774  001062                          BNE     15$             ;;NO, RETURN TO USER
137776  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
140004  001456                          BEQ     15$             ;;BRANCH IF YES
140006  104401  140477                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
140012  104401  140504          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
140016  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
140022  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
140024  104401  140515                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
140030  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
140032  005046                          CLR     -(SP)           ;;THE NEW SWR
140034  105777  041104          7$:     TSTB    @#TKS           ;;CHAR THERE?
140040  100375                          BPL     7$              ;;IF NOT TRY AGAIN
140042  117746  041100                  MOVB    @#TKB,-(SP)     ;;PICK UP CHAR
140046  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
140052  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
```

TTY INPUT ROUTINE

```
140056  001005                          BNE     10$             ;;BRANCH IF NOT
140060  104401  140472                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
140064  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
140070  000757                          BR      19$             ;;LET'S TRY IT AGAIN
140072  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
140076  001022                          BNE     16$             ;;BRANCH IF NO
140100  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
140104  001403                          BEQ     11$             ;;BRANCH IF YES
140106  016677  000002  041024          MOV     2(SP),@SWR      ;;SAVE NEW SWR
140114  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
140120  104401  001175          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
140124  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
140132  001003                          BNE     15$             ;;BRANCH IF NOT
140134  012777  000100  041002          MOV     #100,@#TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
140142  000002          15$:    RTI                             ;;RETURN
140144  004737  137146          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
140150  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
140154  002420                          BLT     18$             ;;BRANCH IF YES
140156  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
140162  003015                          BGT     18$             ;;BRANCH IF YES
140164  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
140170  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
140174  001403                          BEQ     17$             ;;BRANCH IF YES
140176  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
140200  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
140202  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
140204  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
140210  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
140214  000707                          BR      7$              ;;GET THE NEXT ONE
140216  104401  001174          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
140222  000720                          BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB
                                ;;**************************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
140224  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
140226  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
140234  105777  040704          1$:     TSTB    @#TKS           ;;WAIT FOR
140240  100375                          BPL     1$              ;;A CHARACTER
140242  117766  040700  000004          MOVB    @#TKB,4(SP)     ;;READ THE TTY
140250  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
140256  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
140264  001013                          BNE     3$              ;;BRANCH IF NO
140266  105777  040652          2$:     TSTB    @#TKS           ;;WAIT FOR A CHARACTER
140272  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
140274  117746  040646                  MOVB    @#TKB,-(SP)     ;;GET CHARACTER
140300  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
140304  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
140310  001366                          BNE     2$              ;;IF NOT DISCARD IT
140312  000750                          BR      1$              ;;YES, RESUME
140314  026627  000004  000021  3$:     CMP     4(SP),#@#XON    ;;IS IT A RANDOM XON?           ;RAN001
140322  001744                          BEQ     1$              ;;BRANCH IF YES                 ;RAN001
140324  026627  000004  000140          CMP     4(SP),#140      ;;IS IT UPPER CASE?
```

TTY INPUT ROUTINE

```
140332  002407                          BLT     4$                      ;;BRANCH IF YES
140334  026627  000004  000175          CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
140342  003003                          BGT     4$                      ;;BRANCH IF YES
140344  042766  000040  000004          BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
140352  000002                  4$:     RTI                             ;;GO BACK TO USER
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
140354  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
140356  012703  140462          1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
140362  022703  140472          2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
140366  101405                          BLOS    4$                      ;;BR IF YES
140370  104410                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
140372  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
140374  122713  000177          10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
140400  001003                          BNE     3$                      ;;SKIP IF NOT
140402  104401  001174          4$:     TYPE    ,$QUES                  ;;TYPE A '?'
140406  000763                          BR      1$                      ;;CLEAR THE BUFFER AND LOOP
140410  111337  140460          3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
140414  104401  140460                  TYPE    ,9$
140420  122723  000015                  CMPB    #15,(R3)+               ;;CHECK FOR RETURN
140424  001356                          BNE     2$                      ;;LOOP IF NOT RETURN
140426  105063  177777                  CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
140432  104401  001176                  TYPE    ,$LF                    ;;TYPE A LINE FEED
140436  012603                          MOV     (SP)+,R3                ;;RESTORE R3
140440  011646                          MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
140442  016666  000004  000002          MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
140450  012766  140462  000004          MOV     #$TTYIN,4(SP)
140456  000002                          RTI                             ;;RETURN
140460    000                   9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
140461    000                           .BYTE   0                       ;;TERMINATOR
140462                          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
140472    136     125     015   $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
140475    012     000
140477    136     107     015   $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
140502    012     000
140504    015     012     123   $MSWR:  .ASCIZ  <15><12>/SWR = /
140507    127     122     040
140512    075     040     000
140515    040     040     116   $MNEW:  .ASCIZ  /  NEW = /
140520    105     127     040
140523    075     040     000
21672                           .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                           ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
140526  011646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
140530  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
140536  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
140540  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
```

READ AN OCTAL NUMBER FROM THE TTY

```
140542  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
140544  104411          1$:     RDLIN                   ;;READ AN ASCIZ LINE
140546  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
140550  005001                  CLR     R1              ;;CLEAR DATA WORD
140552  005002                  CLR     R2
140554  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
140556  001412                  BEQ     3$              ;;IF ZERO GET OUT
140560  006301                  ASL     R1              ;;*2
140562  006102                  ROL     R2
140564  006301                  ASL     R1              ;;*4
140566  006102                  ROL     R2
140570  006301                  ASL     R1              ;;*8
140572  006102                  ROL     R2
140574  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
140600  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
140602  000764                  BR      2$              ;;LOOP
140604  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
140606  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
140612  010237  140626          MOV     R2,$HIOCT
140616  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
140620  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
140622  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
140624  000002                  RTI                     ;;RETURN
140626  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
21673                   .SBTTL  TRAP DECODER
                        ;;**********************************************************************
                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                        ;*GO TO THAT ROUTINE.
140630  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
140632  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
140636  005740                  TST     -(R0)           ;;BACKUP BY 2
140640  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
140642  006300                  ASL     R0              ;;POSITION FOR INDEXING
140644  016000  140664          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
140650  000200                  RTS     R0              ;;GO TO ROUTINE
                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
140652  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
140654  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
140662  000002                  RTI                     ;;RESTORE THE PSW
                        .SBTTL  TRAP TABLE
                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                        ;*BY THE "TRAP" INSTRUCTION.
                        ;       ROUTINE
                        ;       -------
140664  140652          $TRPAD: .WORD   $TRAP2
140666  136734                  $TYPE   ;;CALL-TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
140670  137314                  $TYPOC  ;;CALL-TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
140672  137270                  $TYPOS  ;;CALL-TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
140674  137330                  $TYPON  ;;CALL-TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
140676  137516                  $TYPDS  ;;CALL-TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
140700  140012                  $GTSWR  ;;CALL-GTSWR     TRAP+6(104406)  GET SOFT-SWR SETTING
140702  137742                  $CKSWR  ;;CALL-CKSWR     TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
140704  140224                  $RDCHR  ;;CALL-RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
140706  140354                  $RDLIN  ;;CALL-RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
140710  140526                  $RDOCT  ;;CALL-RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
```

POWER DOWN AND UP ROUTINES

```
 21674                                  .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
 140712  012737  141052  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
 140720  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
 140726  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 140730  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 140732  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 140734  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 140736  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 140740  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 140742  017746  040172                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
 140746  010637  141056                MOV     SP,$SAVR6       ;;SAVE SP
 140752  012737  140764  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
 140760  000000                        HALT
 140762  000776                        BR      .-2             ;;HANG UP
                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
 140764  012737  141052  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
 140772  013706  141056                MOV     $SAVR6,SP       ;;GET SP
 140776  005037  141056                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
 141002  005237  141056          1$:   INC     $SAVR6          ;;WAIT FOR THE INC
 141006  001375                        BNE     1$              ;;OF  WORD
 141010  012677  040124                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
 141014  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
 141016  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
 141020  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 141022  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 141024  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 141026  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 141030  012737  140712  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
 141036  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
 141044  104401                        TYPE                    ;REPORT THE POWER FAILURE
 141046  141060          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
 141050  000002                        RTI
 141052  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 141054  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
 141056  000000          $SAVR6: 0                       ;;PUT THE SP HERE
 141060     015     012     120  $POWER: .ASCIZ  <15><12>"POWER"
 141063     117     127     105
 141066     122     000
                                        .EVEN
 21676      000001                      .END
```

SYMBOL TABLE

```
ABASE = 000000
ABOEXT  106634
ABORT   136326
ABORTC  136360
ABORTE  136424
ABORTI  105154
ABORTR  105216
ABORTZ  136450
ABORT0  043776
ABORT7  044164
ABROUT  106466
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACTCHS  002724
AC0    =%000000
AC1    =%000001
AC2    =%000002
AC3    =%000003
AC4    =%000004
AC5    =%000005
AC6    =%000006
AC7    =%000007
ADDLSB  111470
ADDT    052160
ADDTRP  135234
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV  = 000000
AENVM = 000000
AFATAL= 000000
ALLCTR  003152
ALLEND  122620
ALR0TS  021512
ALR1TS  021570
ALR2TS  021646
ALR3TS  021724
ALR4TS  022002
ALR5TS  022060
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
A126    025226
BA      002670
BACDAT  103126
BA2     002710
BCR   = 177524
BCSR  = 177520
BDR   = 177524
BFA     052212
BFAC1   052000
BFAC2   052020
BFAC3   052040
BFAC4   052060
BFAC5   052102
BFAE    052252
BFB     052250
BGNTLP  111006
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BNO     043726
BN1     044116
BPTOA   030640
BPTOB   030646
BPTVEC= 000014
BTER    051342
BTEXP   003100
BTGO    051142
BTRES   003110
BTTST   051606
BTTSTE  051634
BYPARR  103434
CCMPAS  003032
CCR   = 177746
CCRTBL  106426
CHECK   015322
CHECK1  015400
CHECK2  015456
CHECK7  053362
CHEC10  053602
CHEC26  056364
CHEC27  056672
CHEC30  057202
CHEC32  057574
CHEC33  060050
CHEK7   053366
CHK10   053614
CHK7    053374
CH10    053606
CKSWR = 104407
CMPRTN  064372
COUNT   003120
CPEREG= 177766
CR    = 000015
CRLF  = 000200
CSR1    002664
CSR12   002704
CSR2    002666
CSR22   002706
CURADD  111504
CURDAT  111476
C121A   023242
C121B   023262
C121C   023302
DAPABO  104670
DATA    002674
DATA2   002714
DATBO   122110
DATI    122440
DATVER  135414
DATVFR  135376
DAT1    135426
DCOUNT  111460
DDISP = 177570
DELAY   114270
DH1     132135
DH105   133076
DH115   133130
DH134   133205
DH24    132421
DH27    132465
DH4     132162
DH41    132530
DH43    132601
DH47    132701
DH5     132247
DH65    132766
DH7     132346
DH72    133032
DISPLA  001142
DISPRE  000174
DMAPAR  121354
DMARD   134766
DMATRN  134704
DPAREN  121414
DSWR  = 177570
DT1     133342
DT105   133620
DT115   133630
DT130   133644
DT134   133652
DT14    133410
DT17    133422
DT24    133434
DT27    133444
DT35    133456
DT4     133350
DT41    133470
DT43    133500
DT47    133514
DT5     133362
DT50    133526
DT51    133540
DT52    133552
DT64    133564
DT65    133576
DT7     133376
DT75    133606
DVDSUB  066372
DVFSUB  065430
D1      052572
D2      052604
D3      052606
D4      052620
D5      052634
D6      052644
D7      052650
EEPAS   003034
EMTOA   030444
EMTOB   030452
EMTSAV  004024
EMTVEC= 000030
EM1     124316
EM10    124570
EM100   130104
EM101   130142
EM102   130175
EM103   130257
EM104   130332
EM105   130402
EM106   130455
EM107   130533
EM11    124611
EM110   130567
EM111   130631
EM112   130707
EM113   130772
EM114   131056
EM115   131077
EM116   131147
EM117   131230
EM12    124633
EM120   131273
EM121   131341
EM122   131437
EM123   131465
EM124   131517
EM125   131611
EM126   131650
EM127   131705
EM13    124661
EM130   131747
EM131   131774
EM132   132024
EM133   132062
EM134   132103
EM14    124710
EM15    124733
EM16    124767
EM17    125034
EM2     124352
EM20    125100
EM21    125133
EM22    125176
EM23    125235
EM24    125311
EM25    125354
EM26    125414
EM27    125463
EM3     124364
EM30    125523
```

H3

SEQ 0447

SYMBOL TABLE

```
EM31    125575  EXITST  111450  HITMIS- 177752  KIPDR1- 172302  MBTOA   033012
EM32    125622  EXPBDT  106446  HMPARR  102724  KIPDR2- 172304  MBTOB   033014
EM33    125663  EXPDAT  111462  HOP10   065642  KIPDR3- 172306  MBTOC   033026
EM34    125725  EXPIR1  034324  HOP11   066604  KIPDR4- 172310  MBTOD   033052
EM35    125765  EXPTBL  103154  HOP12   067612  KIPDR5- 172312  MBTOE   033054
EM36    126013  EXPWDT  106436  HOP13   070610  KIPDR6- 172314  MBTOF   033066
EM37    126057  FINNOP  021232  HOP14   071650  KIPDR7- 172316  MBT1    051120
EM4     124376  FIN1    052334  HOP15   073310  KMCR  - 177734  MBT2    051122
EM40    126121  FIN10   053654  HOP16   073610  LASTCH  113424  MBT2A   051134
EM41    126166  FIN11   053750  HOP17   074242  LCDSUB  076532  MBT8    051344
EM42    126252  FIN116  022722  HOP18   076242  LCFSUB  076130  MBT8A   051374
EM43    126300  FIN117  023036  HOP19   076644  LDPARS  134252  MBT8B   051416
EM44    126331  FIN120  023160  HOP20   100234  LDPDRS  134302  MBT8C   051440
EM45    126371  FIN121  023554  HOP21   101270  LEDS    120332  MBT8D   051462
EM46    126453  FIN122  024024  HOP22   101716  LF    - 000012  MBT8E   051506
EM47    126543  FIN125  025166  HOP44   064510  LKS   - 177546  MBT8F   051532
EM5     124436  FIN126  026226  HT    - 000011  LKSFL   002722  MBT8FG  051546
EM50    126630  FIN127  026666  ILAOA   030740  LKSINT  134376  MBT8I   051572
EM51    126653  FIN13   054134  ILBOB   030746  LOOP    004724  MB66    011754
EM52    126705  FIN130  027556  ILL     053034  LOOPIN  003154  MCB44   011620
EM53    126743  FIN14   054264  ILLBOA  031034  LOST    052320  MCLRD   074276
EM54    126777  FIN15   054452  ILLBOB  031042  LOWADD  003016  MCLRI   074354
EM55    127035  FIN16   054616  ILLOP1  052670  LSTADD  111502  MCMPD   064210
EM56    127071  FIN17   055000  ILLOP2  052744  LXPSUB  100014  MCTSCC  016046
EM57    127115  FIN2    052430  INITMM  134110  MACCC   016176  MDAO    010372
EM6     124471  FIN20   055152  INQ22   122762  MACE    051302  MDCCC   015764
EM60    127147  FIN21   055342  INTERR  105426  MACO    051172  MDDSUB  073044
EM61    127215  FIN22   055506  INTRPC  105372  MACOA   051176  MDFSUB  071374
EM62    127244  FIN23   055626  IOTOA   030346  MAC1    051206  MDIVD   065642
EM63    127315  FIN24   056002  IOTOB   030354  MAC2    051222  MDIVF   064510
EM64    127346  FIN26   056462  IOTVEC- 000020  MAC3    051236  MDMO    010346
EM65    127405  FIN27   056770  IOXXX   033500  MAC4    051252  MDM27   010456
EM66    127455  FIN30   057300  KDPAR0- 172360  MAC5    051266  MDSO    010422
EM67    127514  FIN31   057420  KDPAR1- 172362  MADCC   016122  MEMK    120750
EM7     124534  FIN32   057672  KDPAR2- 172364  MAIREG- 177750  MEMQ    121000
EM70    127551  FIN33   060146  KDPAR3- 172366  MALCC   016750  MEMTO   030400
EM71    127602  FIN4    052750  KDPAR4- 172370  MARCC   017030  MET     032422
EM72    127636  FIN5    053040  KDPAR5- 172372  MASK    003160  META    032460
EM73    127674  FIN6    053214  KDPAR6- 172374  MA11    010774  METB    032470
EM74    127720  FIN7    053434  KDPAR7- 172376  MA55    011700  METD    032500
EM75    127751  FLAG    003030  KDPDR0- 172320  MBB11   011072  METF    032510
EM76    130002  FLO     003062  KDPDR1- 172322  MBB22   011354  METO    032530
EM77    130052  FLOAT   003052  KDPDR2- 172324  MBCCC   015644  METOA   032574
ENDHRT  103116  FMPARR  102534  KDPDR3- 172326  MBC00   010576  METOB   032604
ENDLUP  111424  FPVEC - 000244  KDPDR4- 172330  MBC11   011152  METOC   032616
ENDMOV  111506  FRSTST  004726  KDPDR5- 172332  MBC22   011442  METOD   032642
ENDTAG  112630  FSTADD  111500  KDPDR6- 172334  MBI00   010516  METOE   032652
ENDTLP  111374  FWDSEQ  111466  KDPDR7- 172336  MBPTO   030574  METOF   032664
ERR     135374  GOODAD  003020  KIPAR0- 172340  MBSCC   015716  MFA     051640
ERRCNT  003022  GPR0TS  005070  KIPAR1- 172342  MBT     032704  MFACU   051636
ERRFP   135372  GPR1TS  005146  KIPAR2- 172344  MBTA    032726  MFSRCM  060150
ERROR - 104000  GPR2TS  005224  KIPAR3- 172346  MBTB    032730  MIALL   033300
ERROUT  102056  GPR3TS  005302  KIPAR4- 172350  MBTCC   015574  MIALLA  033324
ERRVEC- 000004  GPR4TS  005360  KIPAR5- 172352  MBTD    032740  MIALLB  033326
ERTYPE  133672  GPR5TS  005436  KIPAR6- 172354  MBTE    051636  MIALLD  033336
EXBAD   111440  GPR6TS  005514  KIPAR7- 172356  MBTF    032750  MIALLF  033346
EXBAD2  112602  GTSWR - 104406  KIPDR0- 172300  MBTO    032762  MIL     033100
```

SYMBOL TABLE

```
MILA    033124  MJSID   033452  MLDDM6  060616  MRTOB   031614  MSXTCC  017120
MILAO   030672  MJSIE   033454  MLDDM7  060702  MRTOC   031626  MS11    011032
MILB    033126  MJSIF   033466  MLDM27  060770  MRTOD   031652  MS22    011304
MILD    033136  MJSR    013676  MLDSUB  070376  MRTOE   031654  MS33    011534
MILF    033146  MJSRA   014014  MLFSUB  067400  MRTOF   031666  MS77    012024
MILLBO  030772  MJSRB   014160  MLS1    074410  MRTS    014762  MTP     031700
MILLO   030204  MJSRC   014326  MLS2    074502  MSB     062616  MTPA    032102
MILLOA  030250  MJSR1   014016  MLS3    074574  MSBCC   016376  MTPAA   032156
MILLOB  030256  MJSR1A  014030  MLS4    074704  MSBCCC  016446  MTPAE   032206
MILO    033160  MJSR1B  014054  MLS5    075000  MSCD    100234  MTPAH   032154
MILOA   033212  MJSR2   013734  MLS6    075110  MSCF    101270  MTPAL   032144
MILOB   033214  MJSR2A  013746  MLS7    075210  MSDF    073610  MTPB    031732
MILOC   033226  MJSR2B  013772  MLXP    076644  MSER  - 177744  MTPF    031752
MILOD   033252  MJSR3   014162  MMARK   017534  MSFD    073310  MTPO    031764
MILOE   033254  MJSR3A  014174  MMODD   071650  MSFDI   074242  MTPOA   032014
MILOF   033266  MJSR3B  014220  MMODF   070610  MSOB    017460  MTPOB   032016
MIOT    032226  MJSR4   014076  MMRLS   012436  MSPAA   007624  MTPOC   032030
MIOTA   032250  MJSR4A  014110  MMR0  - 177572  MSPAU   027556  MTPOD   032054
MIOTB   032252  MJSR4B  014134  MMR1  - 177574  MSPB    005716  MTPOE   032056
MIOTD   032262  MJSR5   014330  MMR2  - 177576  MSPBB   007710  MTPOF   032070
MIOTF   032272  MJSR5A  014342  MMR3  - 172516  MSPC    005742  MTPQ    031742
MIOTO   030302  MJSR5B  014366  MMU     135066  MSPD    005772  MTPR    031730
MITO    032304  MJSR6   014244  MMULD   067612  MSPEO   006030  MTRPO   030476
MITOA   032334  MJSR6A  014256  MMULF   066604  MSPF    006072  MTRY    027720
MITOB   032336  MJSR6B  014302  MMUTRP  135240  MSPG    006134  MTRYA   027774
MITOC   032350  MJSR7   014412  MMVCC   015530  MSPH    006172  MTRYB   030014
MITOD   032374  MJSR7A  014424  MMVEC  - 000250  MSPI    006236  MTRYM   030046
MITOE   032376  MJSR7E  014450  MM11    010664  MSPJ    006276  MTSO    015530
MITOF   032410  MJU1    012756  MM22    011232  MSPL    006416  MTT     031156
MJ      012704  MJU1A   012770  MNCCCC  016260  MSPM    006464  MTTA    031212
MJP     013536  MJU2    012720  MNGOP   062106  MSPN    006542  MTTB    031214
MJP17   013546  MJU2A   012732  MNNRM1  061062  MSPO    006610  MTTD    031224
MJP27   013570  MJU2B   012742  MNNRM2  061302  MSPP    006714  MTTE    031234
MJP27A  013602  MJU3    013002  MNNRM3  061434  MSPQ    007012  MTTR    031340
MJP37   013654  MJU3A   013014  MNNRM4  061556  MSPR    007120  MTTRA   031404
MJP37A  013666  MJU3B   013024  MNRM    062740  MSPS    007206  MTTRB   031406
MJP67   013612  MJU4    013076  MODE1   046014  MSPT    007254  MTTRC   031420
MJP67A  013624  MJU4A   013110  MODE2   047374  MSPU    007272  MTTRD   031456
MJP67B  013640  MJU4B   013122  MODGAR  071640  MSPV    007372  MTTRE   031460
MJP77   013642  MJU5    013036  MRLB1   012164  MSPVO   007336  MTTRF   031472
MJP77E  013676  MJU5A   013050  MRLCC   016532  MSPX    007436  MTTS    031246
MJRA    014454  MJU5B   013062  MRLO    012112  MSPY    007506  MTTSA   031302
MJR27   014510  MJU6    013136  MRL2    012244  MSPZ    007554  MTTSB   031306
MJR27A  014544  MJU6A   013150  MRL3    012322  MSPO    005674  MTTSD   031316
MJR27B  014564  MJU7    013170  MRL4    012376  MSTB3   007760  MTTSE   031326
MJR37   014646  MJU7E   013202  MRL6    012500  MSTO    031062  MTTSQ   031304
MJR37A  014702  MJ2     012754  MRL7    012536  MSTOE   031136  MUVAD   063050
MJR6A   014642  MJ5     013134  MRRB1   012656  MSTOEE  031144  MXDF1   061666
MJR6B   014644  MJ7     013164  MRRCC   016612  MST4    010032  MXOR    017400
MJR67   014566  MLCD    076242  MRRO    012634  MST4B   010070  MXRCC   017170
MJR67A  014622  MLCF    075424  MRT     031504  MST5    010130  M2      004752
MJR77   014722  MLDC    063642  MRTA    031526  MST5B   010162  M3      004766
MJR77A  014756  MLDC2   075324  MRTB    031530  MST6    010224  M4      005006
MJSI    033360  MLDDM2  060220  MRTE    031540  MST7    010272  M5      005026
MJSIA   033412  MLDDM3  060306  MRTF    031550  MSW37   012600  M6      005046
MJSIB   033414  MLDDM4  060422  MRTO    031562  MSXP    101432  NEWADD  003026
MJSIC   033426  MLDDM5  060526  MRTOA   031612  MSXT    017262  NEWDAT  111474
```

J3

SEQ 0449

SYMBOL TABLE

```
NOABRT  136464
NOTOK   135316
NULL  = 000000
NXMFIN  047616
NXMPAR  106670
NXMTRP  047224
ODDXX   033600
OK      135274
OKAY7   043036
OKAY7A  043046
OKA7    043024
OK1     135322
OK7     043006
ONDQ22  134630
PAR     135210
PARAD1  045644
PARAD2  047256
PARVA1  045676
PARVA2  047310
PARVA3  047426
PAR1    135212
PCR   = 177522
PDR     135166
PDR1    135170
PHY1    045762
PIR   = 177772
PIRQ  = 177772
PIRQNX  034202
PIRQT   123252
PIRQVE= 000240
PIRRTN  034502
PIRTBL  034304
PIRTEX  034204
PIRXXX  034146
PIR1    034204
PIR2    034324
PIR2EX  034432
PIR3    034432
PIR3EX  034526
PIR4    034526
PIR5    034604
PIR5EX  035014
PIR6    035014
PIR6EX  035116
PITBL1  034410
PITBL2  035000
PI1     034454
PI2     034466
PI3     034470
PLF0    043654
PLF1    044046
POLY  = 120001
PROCNT  023106
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PSWBTS  005576
PWRVEC= 000024
Q22EN   003002
Q22INT  134664
Q22SIZ  134404
RAMPAR  134326
RBUF  = 177562
RCSR  = 177560
RDCHR = 104410
RDLIN = 104411
RDOCT = 104412
RECDAT  111464
RECDST  003142
RECFEC  003122
RECST   003132
RESVEC= 000010
RET1    015246
RET2    015334
RET3    015412
RITEDA  111472
RTSE    015034
RTS1    015002
RTS6    015012
RXXX    034020
R6    =%000006
R7    =%000007
SAVBR   002730
SAVMR0  003042
SAVMR1  003044
SAVMR2  003046
SAVPCR  002726
SAVPOS  003156
SAVSUP  003036
SAVSWR  003050
SAVUSE  003040
SAV30   004112
SAV32   004114
SCDSUB  101054
SCOPE = 000004
SDFSUB  074040
SDPAR0= 172260
SDPAR1= 172262
SDPAR2= 172264
SDPAR3= 172266
SDPAR4= 172270
SDPAR5= 172272
SDPAR6= 172274
SDPAR7= 172276
SDPDR0= 172220
SDPDR1= 172222
SDPDR2= 172224
SDPDR3= 172226
SDPDR4= 172230
SDPDR5= 172232
SDPDR6= 172234
SDPDR7= 172236
SEQ     003072
SFDSUB  073504
SIMGUA  002702
SIPAR0= 172240
SIPAR1= 172242
SIPAR2= 172244
SIPAR3= 172246
SIPAR4= 172250
SIPAR5= 172252
SIPAR6= 172254
SIPAR7= 172256
SIPDR0= 172200
SIPDR1= 172202
SIPDR2= 172204
SIPDR3= 172206
SIPDR4= 172210
SIPDR5= 172212
SIPDR6= 172214
SIPDR7= 172216
SIXBIT  112774
SLEND   120134
SLOC00  003012
SLOC01  003014
SPAU1   027574
SPAU2   027616
SPAU3   027636
SPAU4   027662
SPAU5   027676
SPAU6   027714
SPS     003074
SPSJ    003076
SR0   = 177572
SR1   = 177574
SR2   = 177576
SR3   = 172516
STACK = 001100
START   004024
STBOT = 001000
STKLMT= 177774
STMOVI  110642
STMOVT  111764
SUBT    052126
SWR     001140
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
SXPSUB  101612
TAB1    003234
TAB10   003364
TAB11   003374
TAB11A  003404
TAB12   003414
TAB13   003424
TAB13B  003434
TAB14   003444
TAB15   003454
TAB16   003464
TAB17   003474
TAB18   003504
TAB2    003244
TAB21   003514
TAB22   003524
TAB23   003534
TAB24   003544
TAB25   003554
TAB26   003564
TAB27   003574
TAB28   003604
TAB29   003614
TAB29A  003624
TAB3    003254
TAB30   003634
TAB31   003644
TAB32   003654
TAB33   003664
TAB34   003674
TAB4    003264
TAB40   003704
TAB41   003714
TAB42   003724
TAB43   003734
TAB45   003744
TAB46   003754
TAB47   003764
TAB47A  003774
TAB48   004004
TAB49   004014
TAB5    003274
TAB5A   003304
TAB6    003314
TAB6A   003324
TAB7    003334
TAB8    003344
TAB9    003354
TAPABO  105034
TA114   021150
TA116   022624
TBITVE= 000014
TB114   021160
TC114   021170
TD114   021200
TEMP    002740
TE102   017740
TE103   017762
TE104   020004
TE105   020026
TE106   020050
TE107   020072
TE110   020114
TE111   020136
TE112   020160
TE113   020402
TE113A  020622
TE114   020774
TE115   022144
TE115A  022354
TE115B  022364
TE115C  022372
TE115D  022400
TE115F  022406
TE116   022410
TE116A  022664
TE116B  022670
TE116C  022702
TE116D  022712
TE117   022722
TE117A  023034
TE120   023040
TE120A  023114
TE121   023170
TE122   023554
TE123   024026
TE124   024246
TE125   024500
TE125A  024740
TE126   025166
TE126A  025234
TE126B  025710
TE127   026226
TE127A  026426
TE130   026670
TE130A  027146
```

SYMBOL TABLE

```
TF114    021210    TSF15    054446    TST20    105722    TS26D1   056412    T124D    024466
TG114    021220    TSF16    054612    TST21    106470    TS26D2   056422    T124E    024470
THRBIT   113242    TSF17    054774    TST22    106700    TS26D3   056432    T124F    024474
TH114    021230    TSF2     052404    TST23    107114    TS26D4   056442    T13FIN   046046
TIMDEL   116116    TSF20    055146    TST24    107452    TS26D5   056452    T14      047140
TIMEOU   052740    TSF21    055336    TST25    110230    TS27D0   056710    T14FIN   047512
TIMOUT   003000    TSF22    055502    TST26    110372    TS27D1   056720    T15      050050
TKVEC =  000060    TSF23    055622    TST27    111506    TS27D2   056730    T15A     050126
TMM16A   050514    TSF24    055776    TST3     102062    TS27D3   056740    T15FIN   050154
TMM16B   050374    TSF31    057414    TST30    112630    TS27D4   056750    UDPAR0=  177660
TMM16C   050424    TSF6     053210    TST31    113002    TS27D5   056760    UDPAR1=  177662
TMM16D   050454    TSF7     053400    TST32    113250    TS30D0   057220    UDPAR2=  177664
TMM16E   050504    TSLOOP   112072    TST33    113570    TS30D1   057230    UDPAR3=  177666
TMM16F   050506    TSMA     043076    TST34    113722    TS30D2   057240    UDPAR4=  177670
TM16A    051100    TSMB     043116    TST35    114306    TS30D3   057250    UDPAR5=  177672
TOUT     135056    TSMC     043126    TST36    114472    TS30D4   057260    UDPAR6=  177674
TPVEC =  000064    TSMMU0   015036    TST37    115000    TS30D5   057270    UDPAR7=  177676
TRAPVE=  000034    TSMMU1   035132    TST4     102162    TS32D0   057612    UDPDR0=  177620
TRPFLG   135370    TSMMU2   035216    TST40    115146    TS32D1   057622    UDPDR1=  177622
TRPOA    030542    TSMMU3   036416    TST41    115450    TS32D3   057642    UDPDR2=  177624
TRPOB    030550    TSMMU4   036716    TST42    115616    TS32D4   057652    UDPDR3=  177626
TRTVEC=  000014    TSMMU5   037646    TST43    116126    TS32D5   057662    UDPDR4=  177630
TRYMA    030106    TSMMU6   040000    TST44    116166    TS33D0   060066    UDPDR5=  177632
TRYMB    030140    TSMMU7   042504    TST45    116264    TS33D1   060076    UDPDR6=  177634
TRYMC    030166    TSMMU8   043132    TST46    116346    TS33D2   060106    UDPDR7=  177636
TSD31    057326    TSMMU9   043334    TST47    116562    TS33D3   060116    UFDFLG   004116
TSEND    112612    TSMM10   044234    TST5     102546    TS33D4   060126    UFDSET=  000001
TSFP1    052252    TSMM11   044616    TST50    116614    TS33D5   060136    UIPAR0=  177640
TSFP10   053434    TSMM12   045054    TST51    117160    TS6DA    053170    UIPAR1=  177642
TSFP11   053654    TSMM13   045442    TST52    117426    TS6DAT   053200    UIPAR2=  177644
TSFP12   053750    TSMM14   046510    TST53    117716    TS7      042754    UIPAR3=  177646
TSFP13   054010    TSMM15   047634    TST54    120134    TS7DA1   053404    UIPAR4=  177650
TSFP14   054134    TSMM16   050154    TST55    120456    TS7DA2   053414    UIPAR5=  177652
TSFP15   054266    TSMM16A  040106    TST56    121032    TS7DA4   053424    UIPAR6=  177654
TSFP16   054454    TSMM16B  040544    TST57    121300    TS7FIN   043130    UIPAR7=  177656
TSFP17   054620    TSMM16C  041304    TST6     102732    TS9FIN   044232    UIPDR0=  177600
TSFP2    052336    TSMM16D  042004    TST60    121452    TYPDS =  104405    UIPDR1=  177602
TSFP20   055002    TSM16A   050250    TST61    121652    TYPE  =  104401    UIPDR2=  177604
TSFP21   055154    TSM16B   050276    TST62    122620    TYPOC =  104402    UIPDR3=  177606
TSFP22   055344    TSM16C   050322    TST63    123076    TYPON =  104404    UIPDR4=  177610
TSFP23   055510    TSM16D   050366    TST64    123262    TYPOS =  104403    UIPDR5=  177612
TSFP24   055630    TSM7     043064    TST65    123362    T10FIN   044616    UIPDR6=  177614
TSFP25   056004    TSM9     043470    TST66    123544    T11FIN   045054    UIPDR7=  177616
TSFP26   056122    TSTADD   003024    TST7     103172    T114     021126    UNXPIR   034430
TSFP27   056464    TSTEND   114304    TS10     044534    T116     022572    UQUIET   004120
TSFP3    052430    TSTLOC   003162    TS10D1   053624    T12FIN   045440    VIREOP   124272
TSFP30   056772    TSTLUP   110724    TS10D2   053634    T122A    023652    VIR1     045730
TSFP31   057302    TST1     004724    TS10D4   053644    T122B    024004    VIR2     047342
TSFP32   057422    TST10    103446    TS11     044734    T123A    024210    VIR3     047460
TSFP33   057674    TST11    103710    TS11D1   053740    T123B    024220    VMKOR    004122
TSFP4    052524    TST12    104164    TS12     045372    T123C    024226    VQBE1    002676
TSFP5    052750    TST13    104364    TS14     047114    T123D    024234    VQBE2    002716
TSFP6    053040    TST14    104700    TS15     050000    T123E    024236    VQPR1    002700
TSFP7    053214    TST15    105044    TS16     051004    T123F    024242    VQPR2    002720
TSF10    053620    TST16    105272    TS16A    050776    T124A    024442    WC       002672
TSF13    054130    TST17    105476    TS1822   046046    T124B    024452    WC2      002712
TSF14    054260    TST2     101716    TS26D0   056402    T124C    024460    WLDTRP   135360
```

L3

SEQ 0451

SYMBOL TABLE

```
XBUF  = 177566        $DDW14  001320        $FILLC  001156        $MTYP1  001231        $TKB    001146
XCBIT   016670        $DDW15  001322        $FILLS  001155        $MTYP2  001235        $TKS    001144
XCSR  = 177564        $DDW2   001270        $GDADR  001120        $MTYP3  001241        $TMP0   001160
XME100  017674        $DDW3   001272        $GDDAT  001124        $MTYP4  001245        $TMP1   001162
XME101  017716        $DDW4   001274        $GET42  135536        $MXCNT  136102        $TN   = 000067
$APTHD  000232        $DDW5   001276        $GTSWR  140012        $NULL   001154        $TPB    001152
$ATYC   136512        $DDW6   001300        $HD   = 000001        $NMTST= 000000        $TPFLG  001157
$ATY1   136466        $DDW7   001302        $HIBTS  000232        $OCNT   137512        $TPS    001150
$ATY3   136474        $DDW8   001304        $HIOCT  140626        $OMODE  137514        $TRAP   140630
$ATY4   136504        $DDW9   001306        $ICNT   001104        $OVER   136060        $TRAP2  140652
$AUTOB  001134        $DEVCT  001210        $ILLUP  141052        $PASS   001206        $TRP  = 000013
$BASE   001254        $DEVM   001256        $INTAG  001135        $PASTM  000240        $TRPAD  140664
$BDADR  001122        $DOAGN  135556        $ITEMB  001114        $POWER  141060        $TSTM   000236
$BDDAT  001126        $DTBL   137722        $LF     001176        $PWRDN  140712        $TSTNM  001102
$BELL   001170        $ENDAD  135546        $LFLG   136731        $PWRMG  141046        $TTYIN  140462
$CDW1   001260        $ENDCT  135514        $LPADR  001106        $PWRUP  140764        $TYPDS  137516
$CDW2   001262        $ENDMG  135565        $LPERR  001110        $QUES   001174        $TYPE   136734
$CHARC  137264        $ENULL  135562        $MADR1  001232        $RDCHR  140224        $TYPEC  137146
$CKSWR  137742        $ENV    001220        $MADR2  001236        $RDLIN  140354        $TYPEX  137266
$CMTAG  001100        $ENVM   001221        $MADR3  001242        $RDOCT  140526        $TYPOC  137314
$CM3  = 000000        $EOP    135446        $MADR4  001246        $RDSZ = 000010        $TYPON  137330
$CM4  = 000002        $EOPCT  135506        $MAIL   001200        $RTNAD  135560        $TYPOS  137270
$CNTLG  140477        $ERFLG  001103        $MAMS1  001230        $SAVR6  141056        $UNIT   001212
$CNTLU  140472        $ERMAX  001115        $MAMS2  001234        $SCOPE  135602        $UNITM  000242
$CPUOP  001226        $ERROR  136104        $MAMS3  001240        $SETUP= 000137        $USWR   001224
$CRLF   001175        $ERRPC  001116        $MAMS4  001244        $STUP = 177777        $VECT1  001250
$DBLK   137732        $ERRTB  001324        $MBADR  000234        $SVLAD  136024        $VECT2  001252
$DDW0   001264        $ERTTL  001112        $MFLG   136730        $SVPC = 000232        $XOFF = 000023
$DDW1   001266        $ESCAP  001166        $MNEW   140515        $SWR  = 167400        $XON  = 000021
$DDW10  001310        $ETABL  001220        $MSGAD  001214        $SWREG  001222        $XTSTR  135622
$DDW11  001312        $ETEND  001324        $MSGLG  001216        $SWRMK= 000300        $$GET4= 000000
$DDW12  001314        $FATAL  001202        $MSGTY  001200        $TESTN  001204        $OFILL  137513
$DDW13  001316        $FFLG   136732        $MSWR   140504        $TIMES  001164        .$X   = 000232

. ABS.  141070      000
        000000      001
ERRORS DETECTED:  10

VIRTUAL MEMORY USED:  64117 WORDS  ( 251 PAGES)
DYNAMIC MEMORY:  19748 WORDS  ( 75 PAGES)
ELAPSED TIME:  00:19:22
APTOT,APTOT/NL:TOC/-SP/CR=ORION.MLB/ML,APTOT.MAC/DS:GBL
```

M3

SEQ 0452

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
ABASE	= 000000	6-454 6-454
ABOEXT	106634	21-17735 21-17745 #21-17747
ABORT	136326	64-21666 #64-21666
ABORTC	136360	64-21666 #64-21666
ABORTE	136424	64-21666 #64-21666
ABORTI	105154	#17-17300 17-17322
ABORTR	105216	17-17286 17-17292 #17-17311
ABORTZ	136450	64-21666 64-21666 #64-21666
ABORTO	043776	7-9305 #7-9421
ABORT7	044164	7-9316 #7-9483
ABROUT	106466	20-17574 #20-17673
ACCC	= ******	7-4167
ACDW1	= 000000	6-454 6-454
ACDW2	= 000000	6-454 6-454
ACPUOP	= 000000	6-454 6-454
ACTCHS	002724	#7-941 *30-18845 *30-18850 30-18860 *30-18874 *30-18879 30-18894 *31-18932
AC0	=%000000	#7-1028 *7-8135 7-8139 7-8140 7-8141 7-8177 *7-8178 7-8179 *7-8189 7-8190 *7-8191 7-8192 *7-8202 7-8203 *7-8212 7-8276 *7-8277 7-8278 *7-8291 7-8292 *7-10497 7-10498 *7-10500 *7-10511 7-10512 *7-10523 7-10524 *7-10525 *7-10527 7-10528 *7-10529 7-10548 *7-10576 7-10577 *7-10584 7-10585 *7-10650 7-10651 *7-10653 7-10654 *7-10662 7-10664 *7-10930 7-10943 *7-10984 *7-10987 7-10993 *7-10998 7-11005 *7-11048 *7-11051 7-11057 *7-11062 7-11069 *7-11110 *7-11111 7-11114 *7-12221 *7-12257 7-12259 *7-12298 7-12300 *7-12341 *7-12344 7-12348 *7-12385 7-12387 *7-12421 7-12423 *7-12455 7-12457 *7-12492 7-12501 *7-12530 *7-12531 7-12538 *7-12546 *7-12547 7-12548 *7-12559 *7-12560 7-12566 *7-12573 *7-12574 7-12576 *7-12606 *7-12609 *7-12610 7-12612 *7-12624 *7-12625 7-12627 *7-12659 *7-12660 7-12662 *7-12673 *7-12674 7-12676 *7-12708 *7-12709 7-12710 *7-12721 *7-12722 7-12723 *7-12750 *7-12751 7-12753 *7-12765 *7-12766 7-12768 *7-12777 *7-12779 7-12781 *7-12790 *7-12791 7-12792 *7-12819 *7-12820 7-12823 *7-12838 *7-12839 7-12845 *7-12868 *7-12869 7-12875 *7-12898 *7-12899 7-12902 *7-12916 *7-12917 7-12920 *7-12934 *7-12935 7-12938 *7-12970 *7-12971 7-12973 *7-12985 *7-12986 7-12988 *7-13019 *7-13020 7-13024 *7-13033 *7-13034 7-13037 *7-13062 *7-13063 7-13066 *7-13082 *7-13084 7-13090 *7-13113 *7-13114 7-13117 *7-13134 *7-13135 7-13141 *7-13164 *7-13165 7-13168 *7-13185 *7-13186 7-13192 *7-13230 *7-13231 7-13233 *7-13249 *7-13250 7-13257 *7-13272 *7-13285 *7-13286 7-13289 *7-13305 *7-13306 7-13309 *7-13384 7-13387 7-13392 *7-13599 *7-13603 7-13631 *7-13766 *7-13770 7-13798 *7-13978 *7-13984 7-14012 *7-14151 *7-14157 7-14185 *7-14362 *7-14370 7-14399 *7-14598 *7-14606 7-14635 *7-14730 7-14735 *7-14836 7-14841 7-14908 *7-14938 7-14940 *7-15288 7-15304 *7-15472 7-15478 *7-15593 7-15599 *7-15821 *7-15826 7-15856 *7-16033 7-16037 *7-16205 7-16209
AC1	=%000001	#7-1029 *7-8136 *7-8141 7-8215 *7-8223 7-8316 *7-10501 *7-10514 7-10515 7-10555 *7-10660 7-10670 *7-10986 7-10987 7-10989 *7-10996 7-10998 7-11001 *7-11050 7-11051 7-11053 *7-11060 7-11062 7-11065 *7-14364 7-14412 *7-14600 7-14648
AC2	=%000002	#7-1030 *7-8137 7-8226 *7-8234 7-8329 *7-10502 *7-10517 7-10518 7-10562 *7-10658 7-10676 *7-10695 7-10696
AC3	=%000003	#7-1031 *7-8138 7-8237 *7-8245 7-8342 *7-10503 *7-10520 7-10521 7-10569 *7-10656 7-10682 *7-10688 7-10689
AC4	=%000004	#7-1032 *7-8139 7-8191 *7-8203 7-8291 *7-10504 *7-10524 7-10525 7-10576 *7-10654 7-10688
AC5	=%000005	#7-1033 *7-8140 7-8178 *7-8190 7-8277 *7-10505 *7-10528 7-10529 7-10584

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		*7-10651	7-10695							
AC6	=#000006	#7-1034	*7-14908							
AC7	=#000007	#7-1035	*7-11142							
ADCC	= ******	7-4136								
ADDLSB	111470	*26-18286	26-18361	26-18366	*26-18376	26-18377	#26-18390	*27-18550	27-18612	27-18617
		*27-18629	27-18630							
ADDT	052160	7-10669	7-10675	7-10681	7-10687	7-10694	#7-10708			
ADDTRP	135234	7-8094	7-8118	#62-21583						
ADDW0	= 000000	6-454	6-454							
ADDW1	= 000000	6-454	6-454							
ADDW10	= 000000	6-454	6-454							
ADDW11	= 000000	6-454	6-454							
ADDW12	= 000000	6-454	6-454							
ADDW13	= 000000	6-454	6-454							
ADDW14	= 000000	6-454	6-454							
ADDW15	= 000000	6-454	6-454							
ADDW2	= 000000	6-454	6-454							
ADDW3	= 000000	6-454	6-454							
ADDW4	= 000000	6-454	6-454							
ADDW5	= 000000	6-454	6-454							
ADDW6	= 000000	6-454	6-454							
ADDW7	= 000000	6-454	6-454							
ADDW8	= 000000	6-454	6-454							
ADDW9	= 000000	6-454	6-454							
ADEVCT	= 000000	6-454	6-454							
ADEVM	= 000000	6-454	6-454							
AENV	= 000000	6-454	6-454							
AENVM	= 000000	6-454	6-454							
AFATAL	= 000000	6-454	6-454							
ALCC	= ******	7-4417								
ALLCTR	003152	#7-1017	*24-18031	*24-18045	*24-18055	*24-18078				
ALLEND	122620	52-20540	#52-20685							
ALR0TS	021512	#7-5220								
ALR1TS	021570	#7-5246								
ALR2TS	021646	#7-5272								
ALR3TS	021724	#7-5298								
ALR4TS	022002	#7-5324								
ALR5TS	022060	#7-5350								
ALTREG	= ******	7-5128								
ALTR1	= ******	7-5245								
ALTR2	= ******	7-5271								
ALTR3	= ******	7-5297								
ALTR4	= ******	7-5323								
ALTR5	= ******	7-5349								
AMADR1	= 000000	6-454	6-454							
AMADR2	= 000000	6-454	6-454							
AMADR3	= 000000	6-454	6-454							
AMADR4	= 000000	6-454	6-454							
AMAMS1	= 000000	6-454	6-454							
AMAMS2	= 000000	6-454	6-454							
AMAMS3	= 000000	6-454	6-454							
AMAMS4	= 000000	6-454	6-454							

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
AMSGAD	= 000000	6-454 6-454
AMSGLG	= 000000	6-454 6-454
AMSGTY	= 000000	6-454 6-454
AMTYP1	= 000000	6-454 6-454
AMTYP2	= 000000	6-454 6-454
AMTYP3	= 000000	6-454 6-454
AMTYP4	= 000000	6-454 6-454
APASS	= 000000	6-454 6-454
APRIOR	= 000000	6-454
APTCSU	= 000040	#64-21667 64-21668
APTENV	= 000001	41-19687 42-19731 45-19942 47-20121 48-20182 49-20261 64-21666 64-21667 #64-21667 64-21668
APTSIZ	= 000200	7-1149 #64-21667
APTSPO	= 000100	64-21667 #64-21667 64-21668
ARCC	= ******	7-4451
ASWREG	= 000000	6-454 6-454
ATESTN	= 000000	6-454 6-454
AUNIT	= 000000	6-454 6-454
AUSWR	= 000000	6-454 6-454
AVECT1	= 000000	6-454 6-454
AVECT2	= 000000	6-454 6-454
A11	= ******	7-2718
A126	025226	#7-6134 7-6142
A55	= ******	7-3012
BA	002670	#7-924 52-20594 52-20630 57-20981 62-21503 62-21514
BACDAT	103126	10-16613 #10-16635
BA2	002710	#7-934 57-20982
BB11	= ******	7-2758
BB22	= ******	7-2875
BCCC	= ******	7-4005
BCR	= 177524	#7-951 40-19648
BCSR	= 177520	#7-953 *8-16337 *8-16348 *9-16437 *9-16440 *9-16477 *9-16480 *9-16542 *9-16554 *10-16609 *10-16632 *11-16751 *11-16778 *12-16848 *12-16871 *13-16936 *13-16944 *13-16952 *13-16960 *14-17016 *14-17038 *15-17124 *15-17161 *16-17216 *16-17228 *17-17295 *17-17304 *18-17382 *18-17391 *19-17447 *19-17472 *20-17578 *20-17591 *20-17597 *20-17621 *20-17625 *20-17648 *21-17737 *21-17749 *22-17799 *22-17828 *23-17903 *23-17951 *24-18027 *24-18080 *25-18126 *25-18127 *25-18129 *25-18130 *26-18241 *26-18276 *26-18283 *26-18382 *27-18494 *27-18532 *27-18536 *27-18636 29-18743 *29-18744 *29-18751 29-18753 *29-18756 29-18758 *29-18765 29-18766 *29-18773 29-18774 *29-18779 29-18780 *29-18786 29-18788 *29-18791 *29-18792 30-18837 *30-18838 *30-18839 *30-18901 30-18902 31-18923 *31-18924 *31-18925 *31-18949 32-19022 *32-19023 *32-19077 33-19129 34-19213 *34-19233 *34-19248 35-19310 35-19315 *35-19316 35-19317 *35-19321 35-19322 *35-19337 36-19381 36-19386 *36-19387 *36-19388 36-19389 *36-19393 36-19394 *36-19403 *36-19405 36-19420 36-19424 *36-19430 37-19464 37-19469 *37-19470 38-19535 38-19538 38-19543 *38-19548 *38-19550 *38-19553 *38-19583 46-20046 *46-20047 *46-20077 *48-20176 *48-20208 *49-20285 *49-20288 *50-20426 50-20431 *50-20435 *50-20440 *50-20450 *50-20451 *52-20559 *52-20562 *52-20579 *52-20582 *52-20603 *52-20606 *52-20636 *52-20639 *52-20677 *52-20680 57-21004 *57-21006 *57-21026 *57-21032 *57-21051 58-21203 *64-21665 *64-21665 *64-21666
BC00	= ******	7-2637
BC11	= ******	7-2790
BC22	= ******	7-2919
BDR	= 177524	#7-952 *48-20193 *48-20207

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
BFA	052212	7-10665	7-10671	7-10677	7-10683	7-10690	7-10697	#7-10714		
BFAC1	052000	7-10666	#7-10669							
BFAC2	052020	7-10672	#7-10675							
BFAC3	052040	7-10678	#7-10681							
BFAC4	052060	7-10684	#7-10687							
BFAC5	052102	7-10691	#7-10694							
BFAE	052252	7-10698	7-10701	#7-10724						
BFB	052250	7-10715	7-10717	7-10719	#7-10721					
BGNTLP	111006	#26-18304	26-18370							
BIT0	= 000001	#5-442								
BIT00	= 000001	#5-442	5-442	7-10180	12-16870	16-17220	19-17448	19-17464	20-17630	20-17640
		43-19793	43-19798	46-20048	46-20049	46-20057	46-20068	46-20069	48-20178	49-20257
		49-20374	49-20388	50-20436	52-20622	57-21005	62-21486	62-21488	62-21500	62-21505
		62-21519	62-21524							
BIT01	= 000002	#5-442	5-442	9-16445	9-16485	23-17902	23-17909	23-17928	23-17934	
BIT02	= 000004	#5-442	5-442	9-16433	9-16539	9-16545	9-16550	11-16755	11-16761	11-16771
		12-16851	12-16861	12-16867	13-16939	13-16955	14-17019	14-17024	14-17029	14-17034
		15-17131	15-17147	16-17223	24-18047	24-18059	24-18072	27-18575	27-18581	27-18588
		42-19737	42-19738	42-19774	43-19793	43-19798	44-19886	44-19891	44-19907	45-19953
		45-19969	45-19983	46-20045	46-20065	46-20071	46-20086	46-20088	47-20125	47-20140
		47-20157	47-20159	49-20289	49-20374	49-20380	52-20563	52-20583	52-20607	52-20640
		52-20681								
BIT03	= 000010	#5-442	5-442	7-6858	9-16476	27-18583	29-18766			
BIT04	= 000020	#5-442	5-442	7-7685	7-9894	7-9908	23-17918	23-17943	24-18024	24-18092
		27-18497	29-18773	29-18774	29-18776	29-18779	29-18780	29-18786	29-18788	29-18791
		49-20275	49-20306	51-20506						
BIT05	= 000040	#5-442	5-442	7-10187	9-16457	30-18838	31-18925	55-20859	62-21410	
BIT06	= 000100	#5-442	5-442	7-1158	7-7721	7-7744	15-17127	15-17142	15-17144	18-17383
		19-17448	19-17464	27-18491	30-18838	31-18924	33-19127	34-19211	34-19218	34-19222
		34-19223	34-19235	34-19245	34-19262	35-19308	36-19379	36-19422	36-19426	36-19429
		37-19462	37-19473	37-19485	38-19533	38-19551	38-19555	38-19563	38-19570	38-19584
		40-19646	44-19839	44-19840	44-19843	44-19844	44-19848	44-19851	44-19859	44-19873
		44-19882	44-19901	45-19966	45-19977	45-19979	45-19996	48-20189	62-21404	64-21664
BIT07	= 000200	#5-442	5-442	7-1189	11-16749	17-17298	20-17582	20-17602	20-17613	26-18238
		30-18838	33-19146	34-19236	34-19242	37-19482	40-19648	46-20038	51-20476	51-20496
		52-20543	53-20717	54-20799	55-20852	56-20890	57-20969	62-21407		
BIT08	= 000400	#5-442	5-442	7-9857	19-17474	25-18127	25-18129	26-18283	26-18382	27-18536
		27-18636	50-20435	50-20440	64-21665					
BIT09	= 001000	#5-442	5-442	14-17015	14-17037	19-17450	25-18126	25-18130	26-18276	46-20047
		49-20363	49-20394	52-20576	62-21423	62-21500	62-21519	64-21665	64-21666	
BIT1	= 000002	#5-442								
BIT10	= 002000	#5-442	16-17218	17-17296	19-17448	19-17464	50-20436	50-20439	51-20493	51-20495
		64-21666								
BIT11	= 004000	#5-442	7-5140	7-5151	7-5152	7-5180	7-5187	7-5194	7-5372	46-20063
		46-20084	64-21665							
BIT12	= 010000	#5-442	32-19011	33-19129	34-19213	35-19310	35-19316	35-19317	35-19321	35-19322
		35-19324	36-19403	36-19405	36-19420	36-19424	37-19464	37-19470	38-19535	52-20598
		52-20671								
BIT13	= 020000	#5-442	36-19381	36-19388	36-19389	36-19393	36-19394	36-19396	46-20063	46-20084
		64-21666								
BIT14	= 040000	#5-442	47-20155	64-21665						
BIT15	= 100000	#5-442	12-16843	12-16847	12-16854	12-16857	23-17915	23-17940	24-18034	24-18036

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		27-18537	46-20063	46-20084	47-20155	49-20390	52-20621			
BIT2	= 000004	#5-442	22-17808	22-17819	22-17829	24-18062				
BIT3	= 000010	#5-442	22-17805	22-17816	22-17825					
BIT4	= 000020	#5-442								
BIT5	= 000040	#5-442	7-1148							
BIT6	= 000100	#5-442	7-1148	47-20142	59-21259					
BIT7	= 000200	#5-442	47-20142	59-21259						
BIT8	= 000400	#5-442	59-21259							
BIT9	= 001000	#5-442								
BIOO	= ******	7-2605								
BNO	043726	7-9304	#7-9401							
BN1	044116	7-9315	7-9326	#7-9464						
BPTO	= ******	7-7011								
BPTOA	030640	7-7018	#7-7024							
BPTOB	030646	7-7020	#7-7027							
BPTVEC	= 000014	#5-442								
BSCC	= ******	7-4034								
BT	= ******	7-7497								
BTCC	= ******	7-3977								
BTER	051342	7-10532	7-10534	7-10536	#7-10539					
BTEXP	003100	#7-1009	7-10495	7-10511	7-10514	7-10517	7-10520	7-10523	7-10527	7-10531
		7-10533	7-10535	7-10537	7-10644	*7-10646	*7-10647	*7-10648	*7-10649	7-10650
		7-10653	7-10656	7-10658	7-10660	7-10662	*7-10703	*7-10704	*7-10705	*7-10706
		*7-10708	*7-10709	*7-10710	*7-10711	7-10714	7-10716	7-10718	7-10720	
BTGO	051142	#7-10500	7-10600							
BTO	= ******	7-7521								
BTRES	003110	#7-1010	7-10496	*7-10512	*7-10515	*7-10518	*7-10521	7-10531	7-10533	7-10535
		7-10537	*7-10548	*7-10555	*7-10562	*7-10569	*7-10577	*7-10585	7-10608	7-10610
		7-10612	7-10614	7-10645	*7-10664	*7-10670	*7-10676	*7-10682	*7-10689	*7-10696
		7-10714	7-10716	7-10718	7-10720					
BTTST	051606	7-10549	7-10556	7-10563	7-10570	7-10578	7-10586	#7-10608		
BTTSTE	051634	7-10609	7-10611	7-10613	#7-10615					
BT1	= ******	7-10477								
BYPARR	103434	11-16747	#11-16783							
B66	= ******	7-3037								
CBIT	= ******	7-4381								
CB44	= ******	7-2982								
CCMPAS	003032	#7-996	*7-1180	*7-1182	*7-1184	*7-1186	7-16272	8-16330	9-16424	9-16531
		10-16599	11-16741	12-16835	13-16920	14-17007	15-17112	16-17209	17-17278	18-17366
		19-17432	20-17567	21-17723	22-17793	23-17891	24-18013	25-18111	26-18232	27-18485
		38-19528	50-20417	51-20470	52-20537	57-20963	57-21057	*57-21059		
CCR	= 177746	#7-954	7-16282	7-16285	*8-16336	*8-16338	8-16339	8-16341	*9-16433	*9-16435
		*9-16476	*9-16493	*9-16539	*9-16541	*11-16749	*11-16777	*13-16935	*13-16951	*14-17015
		*14-17037	*15-17125	*15-17127	*15-17142	*15-17144	*15-17154	*15-17159	*16-17218	*16-17220
		*17-17296	*17-17298	*18-17383	*18-17385	*19-17448	*19-17450	*19-17462	*19-17464	*19-17466
		*19-17471	*19-17474	*20-17580	*20-17582	*20-17600	*20-17602	*20-17611	*20-17613	*20-17628
		*20-17630	*20-17638	*20-17640	*20-17650	*21-17738	*21-17740	*21-17747	*23-17902	*23-17909
		*23-17928	*23-17934	*24-18029	*24-18047	*24-18059	*24-18072	*24-18089	*25-18117	*26-18245
		*26-18278	*27-18498	*27-18529	*49-20266	*49-20363	*49-20394	*50-20436	*50-20439	*50-20449
		*51-20493	*51-20495	*51-20496	*51-20510	*52-20576	*52-20578	58-21190		
CCRTBL	106426	20-17576	20-17595	20-17624	#20-17654					
CHECK	015322	7-3871	#7-3892							

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
CHECK1	015400	7-3893	#7-3912							
CHECK2	015456	7-3913	#7-3932							
CHECK7	053362	7-10990	7-10994	7-11002	7-11006	#7-11012				
CHEC10	053602	7-11054	7-11058	7-11066	7-11070	#7-11076				
CHEC26	056364	7-11721	7-11730	7-11739	7-11750	7-11759	#7-11770			
CHEC27	056672	7-11829	7-11843	7-11857	#7-11866					
CHEC30	057202	7-11925	7-11939	7-11953	#7-11962					
CHEC32	057574	7-12062	7-12071	7-12080	#7-12089					
CHEC33	060050	7-12144	7-12153	7-12162	#7-12171					
CHEK7	053366	#7-11013	7-11017							
CHK10	053614	7-11078	#7-11081							
CHK7	053374	7-11014	#7-11017							
CH10	053606	#7-11077	7-11081							
CKSWR	= 104407	64-21665	64-21666	64-21666	#64-21673					
CLRD	= ******	7-14923								
CLRI	= ******	7-14957								
CMPD	= ******	7-13324								
CMPRTN	064372	7-13338	7-13345	7-13352	7-13358	#7-13378				
COUNT	003120	#7-1011	*7-6799	*7-6805	*7-6806	*7-6812	*7-6818	*7-6819	7-12271	7-12316
		7-12359	7-12398	7-12431	7-12468	7-12507	7-12540	7-12555	7-12568	7-12582
		7-12619	7-12634	7-12668	7-12683	7-12713	7-12725	7-12760	7-12770	7-12784
		7-12795	7-12830	7-12858	7-12888	7-12909	7-12927	7-12945	7-12980	7-12995
		7-13027	7-13039	7-13073	7-13103	7-13124	7-13154	7-13175	7-13205	7-13240
		7-13264	7-13296	7-13316	7-13403	7-13639	7-13806	7-14020	7-14193	7-14407
		7-14416	7-14643	7-14652	7-14748	7-14876	7-14944	7-15306	7-15487	7-15608
		7-15865	7-16073	*63-21646	*63-21651	*63-21652				
CPEREG	= 177766	#7-967	*7-6844	7-6858	*7-6866	*7-6880	*7-6890	*7-6899	*7-6906	*7-6917
		*7-6931	*7-6942	*7-6955	*7-6967	*7-6982	*7-6990	*7-7005	*7-7015	*7-7028
		*7-7040	*7-7056	*7-7676	7-7685	*7-7694	*7-7711	7-7721	*7-7730	7-7744
		*7-7756	*7-7768	7-7781	*7-7784	*7-7798	*7-8090	*7-8110	*7-9963	*7-10181
		7-10187	*7-10197							
CPUTST	= ******	7-1194								
CR	= 000015	#5-442	64-21668	64-21668						
CRLF	= 000200	#5-442	7-1162	7-1162	64-21668	64-21668				
CSR1	002664	#7-922	51-20479	52-20546	52-20596	52-20632	53-20719	53-20724	54-20801	54-20809
		55-20854	56-20913	57-20987	57-20990	57-20997	57-20999	62-21447	62-21472	62-21502
		62-21517	62-21521							
CSR12	002704	#7-932	56-20892	56-20910	57-20972	57-20988	57-20992	57-20996	57-21000	62-21451
CSR2	002666	#7-923	52-20598	52-20671	53-20739	53-20768	54-20816	55-20859	55-20861	55-20864
		56-20912	57-20979	62-21500	62-21505	62-21506	62-21519	62-21524	62-21525	
CSR22	002706	#7-933	56-20916	57-20980						
CTSCC	= ******	7-4095								
CURADD	111504	*24-18030	24-18035	24-18037	24-18039	*24-18040	*24-18050	*24-18051	24-18057	24-18060
		24-18064	24-18068	*24-18071	*24-18076	*26-18302	26-18309	26-18323	26-18330	26-18332
		26-18333	26-18337	*26-18357	*26-18361	26-18362	*26-18364	*26-18366	26-18367	*26-18369
		26-18381	#26-18396	*27-18562	27-18566	27-18573	27-18579	27-18586	27-18591	*27-18608
		*27-18612	27-18613	*27-18615	*27-18617	27-18618	*27-18620	27-18634		
CURDAT	111476	*26-18292	26-18322	26-18324	*26-18343	*26-18347	#26-18393	*27-18551	27-18572	*27-18593
C121A	023242	7-5662	#7-5665							
C121B	023262	7-5667	#7-5670							
C121C	023302	7-5672	#7-5675							
DAPABO	104670	15-17118	#15-17165							

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
DATA	002674	#7-926	52-20657	52-20674	57-20985	62-21497				
DATA2	002714	#7-936	57-20986							
DATBO	122110	52-20584	#52-20589							
DATI	122440	52-20623	#52-20651							
DATVER	135414	7-12270	7-12315	7-12358	7-12397	7-12430	7-12467	7-12506	7-12539	7-12554
		7-12567	7-12581	7-12618	7-12633	7-12667	7-12682	7-12712	7-12724	7-12759
		7-12769	7-12783	7-12794	7-12829	7-12857	7-12887	7-12908	7-12926	7-12944
		7-12979	7-12994	7-13026	7-13038	7-13072	7-13102	7-13123	7-13153	7-13174
		7-13204	7-13239	7-13263	7-13295	7-13315	7-13402	7-13805	7-14192	7-14642
		7-14651	7-14747	7-14943	7-15305	7-15607	7-15864	#63-21649		
DATVFR	135376	7-13638	7-14019	7-14406	7-14415	7-14875	7-15486	7-16072	#63-21644	
DAT1	135426	63-21647	#63-21652	63-21655						
DAO	= ******	7-2535								
DCCC	= ******	7-4061								
DCOUNT	111460	*7-7973	*7-7991	*24-18056	*24-18073	*26-18303	26-18304	26-18328	26-18341	26-18351
		*26-18356	#26-18386	*27-18561	27-18594	27-18599	*27-18607	27-18621		
DDISP	= 177570	#5-442	6-454	7-1149						
DELAY	114270	32-19044	32-19073	#32-19084						
DH1	132135	7-460	7-465	7-470	7-535	7-540	7-545	7-560	7-565	7-580
		7-585	7-590	7-595	7-605	7-610	7-615	7-625	7-635	7-640
		7-670	7-675	7-680	7-685	7-690	7-695	7-700	7-705	7-710
		7-715	7-725	7-730	7-735	7-740	7-750	7-755	7-765	7-770
		7-775	7-780	7-785	7-790	7-795	7-810	7-815	7-835	7-845
		7-850	7-855	7-860	7-865	7-870	7-875	7-880	7-885	7-890
		7-895	7-900	7-905	7-910	#58-21159				
DH105	133076	7-800	7-805	7-825	7-830	#58-21181				
DH115	133130	7-840	#58-21183							
DH134	133205	7-915	#58-21185							
DH24	132421	7-555	#58-21167							
DH27	132465	7-570	7-575	#58-21169						
DH4	132162	7-475	7-515	7-530	7-600	7-660	7-665	7-760	7-820	#58-21161
DH41	132530	7-620	#58-21171							
DH43	132601	7-630	#58-21173							
DH47	132701	7-645	7-650	7-655	#58-21175					
DH5	132247	7-480	7-485	7-520	7-525	7-550	#58-21163			
DH65	132766	7-720	#58-21177							
DH7	132346	7-490	7-495	7-500	7-505	7-510	#58-21165			
DH72	133032	7-745	#58-21179							
DISPLA	001142	#6-454	*7-1149	*7-1149	64-21665	64-21666				
DISPRE	000174	#5-444	7-1149							
DIVD	= ******	7-13656								
DIVF	= ******	7-13413								
DMAPAR	121354	50-20425	#50-20435							
DMARD	134766	52-20656	52-20670	#62-21514						
DMATRN	134704	51-20498	52-20558	52-20577	52-20597	52-20633	#62-21497			
DM0	= ******	7-2517								
DM27	= ******	7-2581								
DPAREN	121414	50-20428	#50-20444							
DSWR	= 177570	#5-442	6-454	7-1149						
DS0	= ******	7-2557								
DT1	133342	7-461	7-466	7-471	7-536	7-541	7-546	7-561	7-566	7-581
		7-586	7-591	7-596	7-606	7-611	7-616	7-626	7-636	7-641

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		7-671	7-676	7-681	7-686	7-691	7-696	7-701	7-706	7-711
		7-716	7-726	7-731	7-736	7-741	7-751	7-756	7-766	7-771
		7-776	7-781	7-786	7-791	7-796	7-811	7-816	7-836	7-846
		7-851	7-856	7-861	7-866	7-871	7-876	7-881	7-886	7-891
		7-901	7-906	7-911	#58-21189					
DT105	133620	7-801	7-806	7-826	7-831	#58-21207				
DT115	133630	7-841	#58-21208							
DT130	133644	7-896	#58-21209							
DT134	133652	7-916	#58-21210							
DT14	133410	7-516	#58-21193							
DT17	133422	7-531	#58-21194							
DT24	133434	7-556	#58-21195							
DT27	133444	7-571	7-576	#58-21196						
DT35	133456	7-601	#58-21197							
DT4	133350	7-476	#58-21190							
DT41	133470	7-621	7-746	#58-21198						
DT43	133500	7-631	#58-21199							
DT47	133514	7-646	7-651	#58-21200						
DT5	133362	7-481	7-486	7-521	7-526	7-551	#58-21191			
DT50	133526	7-656	#58-21201							
DT51	133540	7-661	#58-21202							
DT52	133552	7-666	#58-21203							
DT64	133564	#58-21204								
DT65	133576	7-721	#58-21205							
DT7	133376	7-491	7-496	7-501	7-506	7-511	#58-21192			
DT75	133606	7-761	7-821	#58-21206						
DVDSUB	066372	7-13670	7-13679	7-13688	7-13696	7-13704	7-13712	7-13720	7-13728	7-13737
		#7-13760								
DVFSUB	065430	7-13427	7-13437	7-13446	7-13454	7-13463	7-13471	7-13479	7-13487	7-13495
		7-13503	7-13511	7-13519	7-13527	7-13536	7-13544	7-13553	7-13562	7-13571
		#7-13593								
D1	052572	#7-10830	7-10844	7-10849						
D2	052604	*7-10833	#7-10834							
D3	052606	#7-10835	7-10857							
D4	052620	#7-10841	7-10874	7-10877						
D5	052634	7-10842	#7-10845							
D6	052644	7-10846	#7-10848							
D7	052650	7-10847	#7-10850							
EEPAS	003034	#7-997								
EMT0	= ******	7-6963								
EMT0A	030444	7-6970	#7-6976							
EMT0B	030452	7-6972	#7-6979							
EMTSAV	004024	#7-1148								
EMTVEC	= 000030	#5-442	*7-1149	*7-1149						
EM1	124316	7-459	#58-21066							
EM10	124570	7-494	#58-21073							
EM100	130104	7-774	#58-21129							
EM101	130142	7-779	#58-21130							
EM102	130175	7-784	#58-21131							
EM103	130257	7-789	#58-21132							
EM104	130332	7-794	#58-21133							
EM105	130402	7-799	#58-21134							

SYMBOL	VALUE	REFERENCES	
EM106	130455	7-804	#58-21135
EM107	130533	7-809	#58-21136
EM11	124611	7-499	#58-21074
EM110	130567	7-814	#58-21137
EM111	130631	7-819	#58-21138
EM112	130707	7-824	#58-21139
EM113	130772	7-829	#58-21140
EM114	131056	7-834	#58-21141
EM115	131077	7-839	#58-21142
EM116	131147	7-844	#58-21143
EM117	131230	7-849	#58-21144
EM12	124633	7-504	#58-21075
EM120	131273	7-854	#58-21145
EM121	131341	7-859	#58-21146
EM122	131437	7-864	#58-21147
EM123	131465	7-869	#58-21148
EM124	131517	7-874	#58-21149
EM125	131611	7-879	#58-21150
EM126	131650	7-884	#58-21151
EM127	131705	7-889	#58-21152
EM13	124661	7-509	#58-21076
EM130	131747	7-894	#58-21153
EM131	131774	7-899	#58-21154
EM132	132024	7-904	#58-21155
EM133	132062	7-909	#58-21156
EM134	132103	7-914	#58-21157
EM14	124710	7-514	#58-21077
EM15	124733	7-519	#58-21078
EM16	124767	7-524	#58-21079
EM17	125034	7-529	#58-21080
EM2	124352	7-464	#58-21067
EM20	125100	7-534	#58-21081
EM21	125133	7-539	#58-21082
EM22	125176	7-544	#58-21083
EM23	125235	7-549	#58-21084
EM24	125311	7-554	#58-21085
EM25	125354	7-559	#58-21086
EM26	125414	7-564	#58-21087
EM27	125463	7-569	#58-21088
EM3	124364	7-469	#58-21068
EM30	125523	7-574	#58-21089
EM31	125575	7-579	#58-21090
EM32	125622	7-584	#58-21091
EM33	125663	7-589	#58-21092
EM34	125725	7-594	#58-21093
EM35	125765	7-599	#58-21094
EM36	126013	7-604	#58-21095
EM37	126057	7-609	#58-21096
EM4	124376	7-474	#58-21069
EM40	126121	7-614	#58-21097
EM41	126166	7-619	#58-21098
EM42	126252	7-624	#58-21099

SYMBOL VALUE REFERENCES

EM43	126300	7-629	#58-21100						
EM44	126331	7-634	#58-21101						
EM45	126371	7-639	#58-21102						
EM46	126453	7-644	#58-21103						
EM47	126543	7-649	#58-21104						
EM5	124436	7-479	#58-21070						
EM50	126630	7-654	#58-21105						
EM51	126653	7-659	#58-21106						
EM52	126705	7-664	#58-21107						
EM53	126743	7-669	#58-21108						
EM54	126777	7-674	#58-21109						
EM55	127035	7-679	#58-21110						
EM56	127071	7-684	#58-21111						
EM57	127115	7-689	#58-21112						
EM6	124471	7-484	#58-21071						
EM60	127147	7-694	#58-21113						
EM61	127215	7-699	#58-21114						
EM62	127244	7-704	#58-21115						
EM63	127315	7-709	#58-21116						
EM64	127346	7-714	#58-21117						
EM65	127405	7-719	#58-21118						
EM66	127455	7-724	#58-21119						
EM67	127514	7-729	#58-21120						
EM7	124534	7-489	#58-21072						
EM70	127551	7-734	#58-21121						
EM71	127602	7-739	#58-21122						
EM72	127636	7-744	#58-21123						
EM73	127674	7-749	#58-21124						
EM74	127720	7-754	#58-21125						
EM75	127751	7-759	#58-21126						
EM76	130002	7-764	#58-21127						
EM77	130052	7-769	#58-21128						
ENDHRT	103116	10-16611	#10-16631						
ENDLUP	111424	#26-18377							
ENDMOV	111506	26-18235	26-18240	26-18260	26-18280	#26-18398			
ENDTAG	112630	27-18488	27-18493	27-18514	27-18533	#27-18638			
ENDTLP	111374	26-18305	#26-18371						
ERR	135374	#63-21623							
ERRCNT	003022	#7-992	*32-19036	*32-19048	58-21210				
ERRFP	135372	#63-21622							
ERROR =	104000	#5-442	7-1223	7-1229	7-1247	7-1251	7-1267	7-1280	7-1292	7-1303
		7-1315	7-1320	7-1325	7-1330	7-1341	7-1346	7-1351	7-1356	7-1367
		7-1372	7-1377	7-1382	7-1393	7-1398	7-1403	7-1408	7-1419	7-1424
		7-1429	7-1434	7-1445	7-1450	7-1455	7-1460	7-1471	7-1476	7-1481
		7-1486	7-1498	7-1503	7-1508	7-1513	7-1533	7-1538	7-1552	7-1558
		7-1572	7-1579	7-1595	7-1604	7-1620	7-1631	7-1647	7-1657	7-1674
		7-1682	7-1700	7-1710	7-1726	7-1736	7-1759	7-1768	7-1779	7-1788
		7-1807	7-1817	7-1838	7-1847	7-1851	7-1867	7-1880	7-1903	7-1911
		7-1915	7-1923	7-1926	7-1950	7-1958	7-1962	7-1988	7-1992	7-1996
		7-2003	7-2007	7-2030	7-2034	7-2039	7-2058	7-2066	7-2082	7-2100
		7-2110	7-2129	7-2134	7-2153	7-2162	7-2183	7-2192	7-2214	7-2221
		7-2242	7-2251	7-2274	7-2288	7-2311	7-2317	7-2343	7-2348	7-2369

SYMBOL CROSS REFERENCE CREF V02

SYMBOL VALUE REFERENCES

7-2374 7-2395 7-2399 7-2417 7-2421 7-2443 7-2449 7-2469 7-2475
7-2499 7-2505 7-2528 7-2547 7-2551 7-2569 7-2575 7-2589 7-2594
7-2598 7-2619 7-2623 7-2628 7-2647 7-2653 7-2660 7-2667 7-2691
7-2699 7-2713 7-2732 7-2751 7-2769 7-2773 7-2779 7-2784 7-2802
7-2809 7-2815 7-2838 7-2844 7-2863 7-2869 7-2903 7-2910 7-2914
7-2932 7-2937 7-2943 7-2950 7-2970 7-2974 7-2994 7-3002 7-3008
7-3026 7-3031 7-3057 7-3062 7-3082 7-3085 7-3104 7-3114 7-3133
7-3144 7-3162 7-3174 7-3194 7-3201 7-3224 7-3246 7-3266 7-3286
7-3362 7-3378 7-3395 7-3412 7-3415 7-3423 7-3430 7-3434 7-3442
7-3446 7-3455 7-3459 7-3469 7-3478 7-3502 7-3524 7-3533 7-3539
7-3551 7-3555 7-3580 7-3588 7-3600 7-3608 7-3619 7-3627 7-3640
7-3648 7-3660 7-3668 7-3680 7-3688 7-3699 7-3707 7-3733 7-3749
7-3766 7-3781 7-3797 7-3801 7-3810 7-3814 7-3827 7-3832 7-3837
7-3842 7-3849 7-3854 7-3859 7-3864 7-3876 7-3881 7-3886 7-3897
7-3902 7-3907 7-3917 7-3922 7-3927 7-3935 7-3940 7-3945 7-3963
7-3971 7-3990 7-3999 7-4018 7-4028 7-4046 7-4055 7-4073 7-4089
7-4114 7-4123 7-4130 7-4148 7-4155 7-4161 7-4178 7-4188 7-4195
7-4212 7-4221 7-4231 7-4239 7-4244 7-4262 7-4271 7-4291 7-4306
7-4322 7-4328 7-4335 7-4341 7-4357 7-4363 7-4369 7-4375 7-4400
7-4403 7-4410 7-4413 7-4427 7-4433 7-4440 7-4446 7-4461 7-4468
7-4474 7-4482 7-4500 7-4511 7-4529 7-4538 7-4547 7-4578 7-4591
7-4602 7-4609 7-4627 7-4637 7-4659 7-4662 7-4666 7-4683 7-4689
7-4692 7-4697 7-4710 7-4716 7-4721 7-4725 7-4736 7-4750 7-4761
7-4773 7-4785 7-4797 7-4809 7-4821 7-4833 7-4845 7-4858 7-4864
7-4870 7-4876 7-4882 7-4888 7-4894 7-4900 7-4906 7-4912 7-4924
7-4930 7-4936 7-4942 7-4948 7-4954 7-4960 7-4966 7-4972 7-4978
7-4988 7-4991 7-4995 7-4999 7-5003 7-5007 7-5012 7-5015 7-5019
7-5023 7-5028 7-5031 7-5035 7-5039 7-5044 7-5047 7-5051 7-5055
7-5078 7-5088 7-5096 7-5101 7-5105 7-5109 7-5113 7-5117 7-5121
7-5142 7-5154 7-5158 7-5162 7-5166 7-5170 7-5174 7-5178 7-5197
7-5201 7-5205 7-5209 7-5213 7-5217 7-5225 7-5230 7-5235 7-5240
7-5251 7-5256 7-5261 7-5266 7-5277 7-5282 7-5287 7-5292 7-5303
7-5308 7-5313 7-5318 7-5329 7-5334 7-5339 7-5344 7-5355 7-5360
7-5365 7-5370 7-5388 7-5392 7-5406 7-5410 7-5424 7-5428 7-5432
7-5499 7-5508 7-5512 7-5521 7-5525 7-5529 7-5574 7-5578 7-5585
7-5589 7-5595 7-5622 7-5629 7-5633 7-5637 7-5641 7-5663 7-5668
7-5673 7-5679 7-5685 7-5689 7-5693 7-5712 7-5719 7-5724 7-5729
7-5750 7-5761 7-5839 7-5843 7-5849 7-5854 7-5862 7-5877 7-5912
7-5916 7-5922 7-5926 7-5934 7-5949 7-5971 7-5975 7-5979 7-5983
7-6003 7-6007 7-6012 7-6017 7-6021 7-6135 7-6140 7-6147 7-6158
7-6162 7-6166 7-6174 7-6178 7-6182 7-6192 7-6197 7-6201 7-6205
7-6213 7-6217 7-6221 7-6237 7-6241 7-6245 7-6249 7-6254 7-6397
7-6401 7-6409 7-6413 7-6417 7-6427 7-6431 7-6435 7-6440 7-6565
7-6569 7-6573 7-6577 7-6586 7-6590 7-6594 7-6598 7-6611 7-6615
7-6619 7-6623 7-6628 7-6803 7-6809 7-6815 7-6822 7-6828 7-6834
7-6855 7-6862 7-6887 7-6896 7-6904 7-6926 7-6952 7-6977 7-7000
7-7025 7-7051 7-7076 7-7104 7-7122 7-7124 7-7128 7-7132 7-7151
7-7155 7-7159 7-7163 7-7182 7-7184 7-7188 7-7197 7-7199 7-7203
7-7219 7-7223 7-7227 7-7246 7-7250 7-7258 7-7262 7-7279 7-7283
7-7287 7-7306 7-7310 7-7317 7-7321 7-7343 7-7348 7-7378 7-7382
7-7386 7-7402 7-7406 7-7413 7-7417 7-7442 7-7446 7-7473 7-7487
7-7505 7-7509 7-7513 7-7531 7-7535 7-7542 7-7546 7-7564 7-7568

SYMBOL CROSS REFERENCE CREF V02

SYMBOL VALUE REFERENCES

7-7572 7-7589 7-7593 7-7600 7-7604 7-7622 7-7626 7-7630 7-7649
7-7653 7-7660 7-7664 7-7684 7-7687 7-7690 7-7693 7-7720 7-7723
7-7726 7-7729 7-7742 7-7746 7-7749 7-7752 7-7755 7-7771 7-7774
7-7777 7-7780 7-7783 7-7803 7-7822 7-7829 7-7877 7-7882 7-7904
7-7906 7-7949 7-7987 7-8008 7-8016 7-8061 7-8067 7-8103 7-8146
7-8150 7-8158 7-8169 7-8183 7-8196 7-8207 7-8218 7-8229 7-8240
7-8250 7-8259 7-8269 7-8281 7-8286 7-8295 7-8300 7-8306 7-8311
7-8319 7-8324 7-8332 7-8337 7-8345 7-8350 7-8356 7-8379 7-8385
7-8391 7-8397 7-8404 7-8410 7-8425 7-8432 7-8473 7-8477 7-8483
7-8487 7-8493 7-8497 7-8503 7-8507 7-8514 7-8518 7-8524 7-8528
7-8565 7-8569 7-8575 7-8579 7-8585 7-8589 7-8595 7-8599 7-8606
7-8610 7-8616 7-8620 7-8635 7-8642 7-8649 7-8656 7-8667 7-8671
7-8675 7-8679 7-8705 7-8711 7-8716 7-8720 7-8724 7-8729 7-8733
7-8742 7-8747 7-8751 7-8756 7-8760 7-8767 7-8772 7-8776 7-8780
7-8807 7-8813 7-8818 7-8822 7-8826 7-8831 7-8835 7-8844 7-8849
7-8853 7-8858 7-8862 7-8871 7-8876 7-8880 7-8885 7-8889 7-8896
7-8901 7-8905 7-8909 7-8937 7-8943 7-8948 7-8952 7-8956 7-8961
7-8970 7-8975 7-8979 7-8984 7-8993 7-8998 7-9002 7-9007 7-9014
7-9019 7-9023 7-9051 7-9057 7-9062 7-9066 7-9070 7-9075 7-9084
7-9089 7-9093 7-9098 7-9107 7-9112 7-9116 7-9121 7-9128 7-9133
7-9137 7-9211 7-9216 7-9220 7-9224 7-9235 7-9240 7-9244 7-9271
7-9276 7-9280 7-9284 7-9346 7-9351 7-9356 7-9360 7-9364 7-9568
7-9572 7-9576 7-9580 7-9623 7-9633 7-9703 7-9707 7-9711 7-9751
7-9770 7-9889 7-9903 7-9916 7-9925 7-9934 7-9943 7-9996 7-10057
7-10060 7-10189 7-10192 7-10195 7-10259 7-10278 7-10292 7-10317 7-10323
7-10329 7-10339 7-10347 7-10355 7-10363 7-10369 7-10374 7-10378 7-10382
7-10386 7-10390 7-10394 7-10398 7-10404 7-10414 7-10418 7-10422 7-10434
7-10439 7-10443 7-10447 7-10451 7-10455 7-10461 7-10539 7-10551 7-10558
7-10565 7-10572 7-10580 7-10588 7-10667 7-10673 7-10679 7-10685 7-10692
7-10699 7-10749 7-10778 7-10794 7-10800 7-10808 7-10814 7-10839 7-10859
7-10865 7-10875 7-10879 7-10883 7-10902 7-10907 7-10914 7-10933 7-10939
7-10946 7-10952 7-10969 7-11015 7-11020 7-11079 7-11084 7-11118 7-11123
7-11147 7-11151 7-11173 7-11178 7-11183 7-11190 7-11214 7-11220 7-11225
7-11232 7-11262 7-11267 7-11272 7-11278 7-11283 7-11290 7-11319 7-11324
7-11330 7-11335 7-11342 7-11372 7-11376 7-11381 7-11387 7-11392 7-11399
7-11428 7-11434 7-11440 7-11445 7-11452 7-11482 7-11488 7-11493 7-11498
7-11503 7-11510 7-11541 7-11547 7-11552 7-11559 7-11589 7-11594 7-11601
7-11631 7-11636 7-11641 7-11646 7-11653 7-11680 7-11686 7-11692 7-11697
7-11717 7-11726 7-11735 7-11746 7-11755 7-11764 7-11773 7-11826 7-11832
7-11839 7-11846 7-11853 7-11860 7-11869 7-11922 7-11928 7-11935 7-11942
7-11949 7-11956 7-11965 7-12017 7-12022 7-12027 7-12035 7-12065 7-12074
7-12083 7-12092 7-12147 7-12156 7-12165 7-12174 7-12226 7-12231 7-12235
7-12262 7-12266 7-12273 7-12303 7-12307 7-12311 7-12318 7-12351 7-12355
7-12361 7-12390 7-12394 7-12400 7-12426 7-12433 7-12460 7-12464 7-12470
7-12498 7-12504 7-12509 7-12535 7-12542 7-12552 7-12557 7-12564 7-12570
7-12579 7-12584 7-12615 7-12621 7-12630 7-12636 7-12665 7-12670 7-12679
7-12685 7-12715 7-12727 7-12756 7-12762 7-12772 7-12786 7-12797 7-12826
7-12832 7-12841 7-12848 7-12853 7-12860 7-12871 7-12878 7-12883 7-12890
7-12905 7-12911 7-12923 7-12929 7-12941 7-12947 7-12976 7-12982 7-12991
7-12997 7-13029 7-13041 7-13069 7-13075 7-13086 7-13093 7-13098 7-13105
7-13120 7-13126 7-13137 7-13144 7-13149 7-13156 7-13171 7-13177 7-13188
7-13197 7-13201 7-13207 7-13236 7-13242 7-13253 7-13260 7-13266 7-13278

SYMBOL CROSS REFERENCE CREF V02

SYMBOL VALUE REFERENCES

```
                        7-13292  7-13298  7-13312  7-13318  7-13395  7-13405  7-13609  7-13616  7-13623
                        7-13634  7-13641  7-13650  7-13776  7-13783  7-13790  7-13801  7-13808  7-13817
                        7-13990  7-13997  7-14004  7-14015  7-14022  7-14032  7-14163  7-14170  7-14177
                        7-14188  7-14195  7-14205  7-14376  7-14383  7-14390  7-14402  7-14409  7-14418
                        7-14427  7-14612  7-14619  7-14626  7-14638  7-14645  7-14654  7-14663  7-14743
                        7-14750  7-14754  7-14848  7-14855  7-14862  7-14871  7-14878  7-14888  7-14913
                        7-14917  7-14946  7-14951  7-14976  7-14980  7-15006  7-15010  7-15014  7-15020
                        7-15046  7-15050  7-15054  7-15060  7-15088  7-15092  7-15096  7-15102  7-15128
                        7-15132  7-15136  7-15142  7-15170  7-15174  7-15178  7-15184  7-15210  7-15214
                        7-15218  7-15224  7-15252  7-15256  7-15260  7-15266  7-15295  7-15300  7-15308
                        7-15482  7-15489  7-15496  7-15603  7-15610  7-15617  7-15834  7-15841  7-15848
                        7-15860  7-15867  7-15877  7-16045  7-16052  7-16059  7-16066  7-16075  7-16085
                        7-16216  7-16221  7-16228  7-16286  7-16292  7-16298  8-16345  9-16447  9-16451
                        9-16455  9-16459  9-16465  9-16469  9-16472  9-16474  9-16487  9-16489  9-16491
                        9-16547  9-16552  10-16627 11-16757 11-16763 11-16767 11-16773 11-16776 12-16853
                        12-16863 12-16869 13-16964 14-17021 14-17026 14-17031 14-17036 15-17122 15-17133
                        15-17137 15-17141 15-17149 15-17153 15-17158 15-17166 16-17225 16-17233 17-17303
                        17-17315 17-17321 17-17324 18-17390 18-17400 18-17405 18-17409 19-17452 19-17457
                        19-17469 20-17587 20-17607 20-17618 20-17635 20-17645 21-17744 21-17757 22-17807
                        22-17811 22-17818 22-17822 22-17827 22-17832 23-17920 23-17923 23-17945 23-17948
                        24-18066 24-18070 25-18133 26-18272 27-18524 28-18683 28-18695 28-18705 29-18755
                        29-18760 29-18769 29-18777 29-18783 29-18790 30-18857 30-18862 30-18865 30-18888
                        30-18896 30-18899 31-18941 31-18947 32-19052 33-19139 33-19157 33-19161 34-19225
                        34-19238 34-19244 34-19247 34-19251 34-19255 34-19269 35-19320 35-19325 35-19333
                        36-19392 36-19397 36-19415 36-19419 36-19428 37-19484 37-19487 38-19574 38-19578
                        38-19582 39-19607 39-19610 39-19613 40-19661 41-19696 41-19700 42-19741 42-19763
                        42-19769 42-19773 43-19800 44-19842 44-19846 44-19850 44-19853 44-19867 44-19892
                        45-19970 45-19984 46-20051 46-20066 46-20072 46-20087 47-20141 47-20145 47-20158
                        49-20383 49-20393 50-20447 51-20499 51-20508 52-20565 52-20568 52-20585 52-20600
                        52-20609 52-20612 52-20642 52-20645 52-20659 52-20673 52-20676 52-20683 53-20744
                        53-20772 54-20821 55-20862 56-20921 57-21024 57-21050 62-21406 62-21409 62-21412
                        62-21532 62-21590 63-21616
ERROUT    102056    7-16280  #7-16302
ERRVEC  = 000004   #5-442    7-1149   *7-1149   *7-1149   *7-1151   *7-1152    33-19134 *33-19135 *33-19136
                   *33-19142 35-19329 *35-19330 *35-19331 *35-19336  40-19654 *40-19655 *40-19656 *40-19664
                    49-20267 *49-20268 *49-20269 *49-20304  49-20347 *49-20348 *49-20349 *49-20362  62-21429
                   *62-21430 *62-21431 *62-21475  64-21665 *64-21665 *64-21665 *64-21665
ERTYPE    133672   #59-21223  64-21666
ET      = ******   7-7425
ETO     = ******   7-7454
EXBAD     111440   26-18326   26-18336  #26-18380
EXBAD2    112602   27-18577   27-18585   27-18590 #27-18633
EXITST    111450   26-18378  #26-18382   27-18511  27-18517
EXPBDT    106446   20-17596  #20-17664
EXPDAT    111462  *7-5070    *7-5080    7-5093    *7-5418    7-5419    7-5426    7-5430    *7-5963   *7-5964
                   7-5981    *7-6228    7-6247    7-6251    #26-18387
EXPIR1    034324   7-7832    #7-7843
EXPTBL    103154   10-16607  #10-16646
EXPWDT    106436   20-17577  #20-17659
FACU    = ******   7-10624
FINNOP    021232   7-5091    #7-5125
FIN1      052334   7-10743   #7-10752
```

M4

SEQ 0465

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
FIN10	053654	7-11074	#7-11100							
FIN11	053750	7-11127	#7-11133							
FIN116	022722	7-5502	#7-5554							
FIN117	023036	7-5593	#7-5598							
FIN120	023160	7-5624	7-5640	#7-5643						
FIN121	023554	#7-5737								
FIN122	024024	7-5655	7-5766	#7-5815						
FIN125	025166	7-5986	7-6024	7-6028	#7-6120					
FIN126	026226	7-6262	#7-6385							
FIN127	026666	7-6444	7-6448	#7-6551						
FIN13	054134	7-11188	#7-11194							
FIN130	027556	7-6632	7-6636	#7-6792						
FIN14	054264	7-11230	#7-11236							
FIN15	054452	7-11288	#7-11294							
FIN16	054616	7-11340	#7-11346							
FIN17	055000	7-11397	#7-11403							
FIN2	052430	7-10769	#7-10781							
FIN20	055152	7-11450	#7-11456							
FIN21	055342	7-11508	#7-11514							
FIN22	055506	7-11557	#7-11563							
FIN23	055626	7-11599	#7-11604							
FIN24	056002	7-11651	#7-11657							
FIN26	056462	7-11768	#7-11802							
FIN27	056770	7-11864	#7-11898							
FIN30	057300	7-11960	#7-11994							
FIN31	057420	7-12033	#7-12039							
FIN32	057672	7-12087	#7-12121							
FIN33	060146	7-12169	#7-12203							
FIN4	052750	7-10855	#7-10886							
FIN5	053040	7-10912	#7-10917							
FIN6	053214	7-10958	#7-10972							
FIN7	053434	7-11010	#7-11036							
FLAG	003030	#7-995	*7-7860	7-7868	*7-7873	*7-8092	*7-8112	*7-8367	*7-8442	*7-8630
		*7-8663	*7-8686	*7-8790	*7-8919	*7-9033	*7-9147	*7-9173	*7-9185	*7-9205
		7-9209	7-9233	*7-9254	*7-9265	7-9269	*7-9297	*7-9337	7-9344	7-9349
		*7-9366	*7-9509	*7-9517	*7-9528	*7-9539	*7-9552	7-9566	*7-9593	*7-9618
		7-9621	*7-9627	7-9631	*7-9635	*7-9644	*7-9655	*7-9662	*7-9669	*7-9676
		*7-9683	*7-9690	7-9701	*7-9731	*7-9981	*7-10206	*7-10305	*7-10408	7-10412
		*7-13337	*7-13344	*7-13351	7-13397	*7-13426	*7-13436	*7-13445	*7-13453	*7-13462
		*7-13470	*7-13478	*7-13486	*7-13494	*7-13502	*7-13510	*7-13518	*7-13526	*7-13535
		*7-13543	*7-13552	*7-13561	*7-13570	7-13607	7-13614	7-13643	*7-13669	*7-13678
		*7-13687	*7-13695	*7-13703	*7-13711	*7-13719	*7-13727	*7-13736	7-13774	7-13781
		7-13810	*7-13836	*7-13844	*7-13852	*7-13860	*7-13868	*7-13876	*7-13884	*7-13892
		*7-13900	*7-13908	*7-13916	*7-13924	*7-13933	*7-13942	*7-13951	7-13988	7-13995
		7-14024	*7-14050	*7-14058	*7-14066	*7-14074	*7-14082	*7-14090	*7-14098	*7-14106
		*7-14115	*7-14123	7-14161	7-14168	7-14197	*7-14223	*7-14232	*7-14241	*7-14250
		*7-14259	*7-14269	*7-14278	*7-14287	*7-14296	*7-14305	*7-14314	*7-14323	*7-14332
		7-14374	7-14381	7-14420	*7-14448	*7-14457	*7-14466	*7-14475	*7-14484	*7-14493
		*7-14502	*7-14511	*7-14520	*7-14529	*7-14539	*7-14548	*7-14558	*7-14568	7-14610
		7-14617	7-14656	*7-14774	*7-14781	*7-14788	*7-14795	*7-14802	*7-14809	7-14846
		7-14853	7-14880	*7-15638	*7-15646	*7-15654	*7-15662	*7-15670	*7-15678	*7-15687
		*7-15696	*7-15705	*7-15714	*7-15723	*7-15731	*7-15739	*7-15748	*7-15757	*7-15765

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		*7-15774	*7-15782	*7-15791	7-15832	7-15839	7-15869	*7-15897	*7-15904	*7-15911
		*7-15918	*7-15925	*7-15932	*7-15939	*7-15946	*7-15953	*7-15960	*7-15967	*7-15974
		*7-15981	*7-15988	*7-15995	*7-16002	7-16043	7-16050	7-16077	*7-16105	*7-16112
		*7-16119	*7-16126	62-21588	*62-21607					
FLO	003062	#7-1005	*7-8142	7-8143	7-8175	7-8213	7-8224	7-8235	7-8246	7-8275
FLOAT	003052	#7-1004	7-8129	7-8134	7-8176	7-8187	7-8200	7-8274	7-8290	7-8315
		7-8328	7-8341							
FMPARR	102534	9-16430	#9-16498							
FPPTST	= ******	7-1045	7-10470	63-21612						
FPTS1	= ******	7-10729								
FPTS10	= ******	7-11039								
FPTS11	= ******	7-11103								
FPTS12	= ******	7-11136								
FPTS13	= ******	7-11156								
FPTS14	= ******	7-11197								
FPTS15	= ******	7-11239								
FPTS16	= ******	7-11297								
FPTS17	= ******	7-11349								
FPTS2	= ******	7-10755								
FPTS20	= ******	7-11406								
FPTS21	= ******	7-11459								
FPTS22	= ******	7-11517								
FPTS23	= ******	7-11566								
FPTS24	= ******	7-11607								
FPTS25	= ******	7-11660								
FPTS26	= ******	7-11702								
FPTS27	= ******	7-11805								
FPTS3	= ******	7-10784								
FPTS30	= ******	7-11901								
FPTS31	= ******	7-11997								
FPTS32	= ******	7-12042								
FPTS33	= ******	7-12124								
FPTS4	= ******	7-10819								
FPTS5	= ******	7-10889								
FPTS6	= ******	7-10920								
FPTS7	= ******	7-10975								
FPVEC	= 000244	#7-1039	*7-12837	*7-12865	*7-12897	*7-13083	*7-13110	*7-13131	*7-13161	*7-13182
		*7-13594	*7-13761	*7-13975	*7-14148	*7-14359	*7-14595	*7-14727	*7-14833	*7-15001
		*7-15041	*7-15083	*7-15123	*7-15165	*7-15205	*7-15247	*7-15287	*7-15466	*7-15587
		*7-15816	*7-16026	*7-16200						
FRSTST	004726	#7-1210								
FSRCM0	= ******	7-12206								
FSTADD	111500	*26-18297	*26-18300	26-18302	26-18357	#26-18394	*27-18556	*27-18559	27-18562	27-18608
FWDSEQ	111466	*26-18285	26-18295	26-18359	26-18371	*26-18373	*26-18375	#26-18389	*27-18549	27-18554
		27-18610	27-18624	*27-18626	*27-18628					
GNS	= ******	5-444	5-444	7-1162	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673
		64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673
		64-21673	64-21673	64-21673	64-21673	64-21673				
GOODAD	003020	#7-991	*24-18048	*24-18049	24-18064	24-18068	*24-18075			
GPR0TS	005070	#7-1309								
GPR1TS	005146	#7-1335								
GPR2TS	005224	#7-1361								

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
GPR3TS	005302	#7-1387
GPR4TS	005360	#7-1413
GPR5TS	005436	#7-1439
GPR6TS	005514	#7-1465
GTSWR	= 104406	7-1162 #64-21673
HITMIS	= 177752	#7-955 7-16294 7-16297 9-16439 9-16479 9-16544 9-16549 10-16608 10-16622 11-16754 11-16760 11-16770 12-16850 12-16860 12-16866 13-16938 13-16954 14-17018 14-17023 14-17028 14-17033 15-17130 15-17146 16-17222 22-17804 22-17815 22-17824 23-17917 23-17942 24-18061 27-18574 27-18580 27-18587 49-20287 52-20561 52-20581 52-20605 52-20638 52-20679
HMPARR	102724	9-16537 #9-16557
HOP10	065642	7-13580 #7-13654
HOP11	066604	7-13746 #7-13821
HOP12	067612	7-13961 #7-14035
HOP13	070610	7-14133 #7-14208
HOP14	071650	7-14342 #7-14432
HOP15	073310	7-14579 #7-14667
HOP16	073610	7-14713 #7-14758
HOP17	074242	7-14817 #7-14892
HOP18	076242	7-15449 #7-15499
HOP19	076644	7-15571 #7-15621
HOP20	100234	7-15801 #7-15881
HOP21	101270	7-16010 #7-16089
HOP22	101716	7-16186 #7-16233
HOP44	064510	7-13363 #7-13408
HT	= 000011	#5-442 64-21668 64-21668
IALL	= ******	7-7613
IL	= ******	7-7555
ILAOA	030740	7-7043 #7-7050
ILBOB	030746	7-7045 #7-7053
ILL	053034	7-10894 #7-10914
ILLA	= ******	7-7036
ILLB	= ******	7-7062
ILLBOA	031034	7-7068 #7-7075
ILLBOB	031042	7-7070 #7-7078
ILLO	= ******	7-6913
ILLOP1	052670	7-10825 #7-10857
ILLOP2	052744	7-10829 #7-10883
ILO	= ******	7-7578
INITMM	134110	12-16840 13-16925 23-17897 24-18022 26-18242 27-18495 49-20270 52-20619 #59-21301
IN022	122762	#53-20754
INTERR	105426	18-17373 18-17379 #18-17397
INTRPC	105372	#18-17387 18-17403
IOT	= ******	7-7370
IOTIME	= ******	7-7673
IOTO	= ******	7-6938
IOTOA	030346	7-6945 #7-6951
IOTOB	030354	7-6947 #7-6954
IOTVEC	= 000020	#5-442 *7-1149 *7-1149
IOXXX	033500	#7-7674
ITO	= ******	7-7392
JMP	= ******	7-3402

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
JP	= ******	7-3514								
JSI	= ******	7-7638								
JSRA	= ******	7-3565	7-3714							
KDPAR0	= 172360	#5-441	26-18288	*26-18290	27-18539	*27-18541				
KDPAR1	= 172362	#5-441								
KDPAR2	= 172364	#5-441								
KDPAR3	= 172366	#5-441								
KDPAR4	= 172370	#5-441								
KDPAR5	= 172372	#5-441								
KDPAR6	= 172374	#5-441								
KDPAR7	= 172376	#5-441								
KDPDR0	= 172320	#5-441								
KDPDR1	= 172322	#5-441								
KDPDR2	= 172324	#5-441								
KDPDR3	= 172326	#5-441								
KDPDR4	= 172330	#5-441								
KDPDR5	= 172332	#5-441								
KDPDR6	= 172334	#5-441								
KDPDR7	= 172336	#5-441								
KIPAR0	= 172340	#5-441	*7-9968							
KIPAR1	= 172342	#5-441								
KIPAR2	= 172344	#5-441								
KIPAR3	= 172346	#5-441								
KIPAR4	= 172350	#5-441								
KIPAR5	= 172352	#5-441	*27-18543	*27-18572	*27-18578	27-18633	*27-18635			
KIPAR6	= 172354	#5-441	*7-9879	*7-9895	*7-9911	*7-9920	*7-9929	*7-9938	*7-9970	*7-10177
		*13-16926	*21-17732	*23-17898	*24-18028	24-18032	*24-18041	24-18042	*24-18044	*24-18052
		*24-18054	*24-18077	*26-18243	*27-18510	*49-20271	*49-20280	49-20296	*52-20620	58-21200
		58-21208								
KIPAR7	= 172356	#5-441								
KIPDR0	= 172300	#5-441	*7-9969	*12-16843	*12-16847	*12-16854	*12-16857	*26-18287	*26-18291	*27-18538
		*27-18542								
KIPDR1	= 172302	#5-441								
KIPDR2	= 172304	#5-441								
KIPDR3	= 172306	#5-441								
KIPDR4	= 172310	#5-441								
KIPDR5	= 172312	#5-441	*27-18537							
KIPDR6	= 172314	#5-441	*7-9973	*24-18034	*24-18036	*52-20621				
KIPDR7	= 172316	#5-441								
KMCR	= 177734	#7-956								
LASTCH	113424	#30-18874								
LCD	= ******	7-15502								
LCDSUB	076532	7-15515	7-15521	7-15527	7-15533	7-15539	7-15545	7-15551	7-15557	7-15563
		#7-15586								
LCF	= ******	7-15315								
LCFSUB	076130	7-15328	7-15334	7-15340	7-15346	7-15352	7-15358	7-15364	7-15370	7-15376
		7-15382	7-15388	7-15394	7-15400	7-15406	7-15412	7-15418	7-15424	7-15430
		7-15436	7-15442	#7-15465						
LDC	= ******	7-13213								
LDC2	= ******	7-15272								
LDDM2	= ******	7-12241								
LDDM3	= ******	7-12279								

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
LDDM4	= 000000	7-12325
LDDM5	= 000000	7-12368
LDDM6	= 000000	7-12405
LDDM7	= 000000	7-12438
LDM27	= 000000	7-12475
LDPARS	134252	59-21302 59-21304 59-21306 59-21308 59-21310 59-21312 #60-21347
LDPDRS	134302	59-21314 59-21316 59-21318 59-21320 59-21322 59-21324 #61-21377
LEDS	120332	48-20186 #48-20211
LF	= 000012	#5-442 64-21668 64-21668
LKS	= 177546	#7-957 33-19137 *33-19146 33-19152 33-19155 *34-19218 *34-19222 34-19223 *34-19235
		34-19236 34-19239 *34-19241 34-19242 34-19245 *34-19262 35-19332 36-19426 *36-19429
		*37-19473 37-19482 37-19485 *38-19555 *38-19570 *38-19584
LKSFL	002722	#7-940 *33-19147 33-19159 *34-19219 34-19227 34-19252 *34-19259 34-19263 34-19266
		*36-19406 36-19409 36-19416 *37-19474 37-19477 *38-19554 38-19559 58-21204 *62-21415
LKSINT	134376	33-19148 34-19220 36-19401 37-19471 38-19544 #62-21415
LOOP	004724	7-1159 7-1190 #7-1192 64-21664
LOOPIN	003154	#7-1018 *10-16604 10-16610 10-16617 *10-16629 *24-18025 *24-18086
LOST	052320	7-10734 7-10736 7-10738 7-10740 #7-10745
LOWADD	003016	#7-990 *24-18021 24-18030 24-18048 *24-18085
LSTADD	111502	*26-18298 *26-18301 26-18337 #26-18395 *27-18557 *27-18560 27-18591
LS1	= 000000	7-14986
LS2	= 000000	7-15026
LS3	= 000000	7-15066
LS4	= 000000	7-15108
LS5	= 000000	7-15148
LS6	= 000000	7-15190
LS7	= 000000	7-15230
LXP	= 000000	7-15624
LXPSUB	100014	7-15639 7-15647 7-15655 7-15663 7-15671 7-15679 7-15688 7-15697 7-15706
		7-15715 7-15724 7-15732 7-15740 7-15749 7-15758 7-15766 7-15775 7-15783
		7-15792 #7-15815
MACCC	016176	#7-4168
MACE	051302	7-10513 7-10516 7-10519 7-10522 #7-10531
MACO	051172	7-10509 #7-10511
MACOA	051176	#7-10512 7-10526 7-10530
MAC1	051206	#7-10514
MAC2	051222	#7-10517
MAC3	051236	#7-10520
MAC4	051252	#7-10523
MAC5	051266	#7-10527
MADCC	016122	#7-4137
MAIREG	= 177750	#7-958 39-19605 39-19608 39-19611 62-21423
MALCC	016750	#7-4418
MARCC	017030	#7-4452
MARKIT	= 000000	7-4672
MASK	003160	#7-1022
MA11	010774	#7-2719
MA55	011700	#7-3013
MBB11	011072	#7-2759
MBB22	011354	#7-2889
MBCCC	015644	#7-4006
MBCOO	010576	#7-2638

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
MBC11	011152	#7-2791								
MBC22	011442	#7-2920								
MBI00	010516	#7-2606								
MBPTO	030574	#7-7012								
MBSCC	015716	#7-4035								
MBT	032704	#7-7498								
MBTA	032726	#7-7505	7-7511							
MBTB	032730	7-7503	#7-7507							
MBTCC	015574	#7-3978								
MBTD	032740	7-7508	#7-7511							
MBTE	051636	7-10604	#7-10620							
MBTF	032750	7-7512	#7-7515							
MBTO	032762	#7-7522								
MBTOA	033012	#7-7531								
MBTOB	033014	7-7527	#7-7533							
MBTOC	033026	7-7534	#7-7537							
MBTOD	033052	#7-7542								
MBTOE	033054	7-7538	#7-7544							
MBTOF	033066	7-7545	#7-7548							
MBT1	051120	#7-10489								
MBT2	051122	#7-10494								
MBT2A	051134	#7-10497	7-10605							
MBT8	051344	7-10538	#7-10542							
MBT8A	051374	7-10545	7-10547	7-10550	#7-10553					
MBT8B	051416	7-10554	7-10557	#7-10560						
MBT8C	051440	7-10561	7-10564	#7-10567						
MBT8D	051462	7-10568	7-10571	#7-10574						
MBT8E	051506	7-10575	7-10579	#7-10582						
MBT8F	051532	7-10583	7-10587	#7-10590						
MBT8FG	051546	7-10593	#7-10595							
MBT8I	051572	7-10599	#7-10602							
MB66	011754	#7-3046								
MCB44	011620	#7-2983								
MCLRD	074276	#7-14932								
MCLRI	074354	#7-14966								
MCMPD	064210	#7-13333								
MCTSCC	016046	#7-4103								
MDA0	010372	#7-2536								
MDCCC	015764	#7-4062								
MDDSUB	073044	7-14449	7-14458	7-14467	7-14476	7-14485	7-14494	7-14503	7-14512	7-14521
		7-14530	7-14540	7-14549	7-14559	7-14569	#7-14594			
MDFSUB	071374	7-14224	7-14233	7-14242	7-14251	7-14260	7-14270	7-14279	7-14288	7-14297
		7-14306	7-14315	7-14324	7-14333	#7-14358				
MDIVD	065642	#7-13665								
MDIVF	064510	#7-13422								
MDM0	010346	#7-2518								
MDM27	010456	#7-2582								
MDS0	010422	#7-2558								
MD1	= ******	7-1202								
MD2	= ******	7-1232								
MD3	= ******	7-1255								
MD4	= ******	7-1270								

F5

SYMBOL	VALUE	REFERENCES	
MD5	= ******	7-1285	
MD6	= ******	7-1296	
MEMK	120750	49-20312	#49-20321
MEMQ	121000	49-20318	#49-20322
MEMTO	030400	#7-6964	
MET	032422	#7-7426	
META	032460	#7-7435	7-7444
METB	032470	7-7431	#7-7440
METD	032500	7-7441	#7-7444
METF	032510	7-7445	#7-7448
METO	032530	#7-7455	
METOA	032574	#7-7466	
METOB	032604	7-7460	#7-7471
METOC	032616	7-7472	#7-7475
METOD	032642	#7-7480	
METOE	032652	7-7476	#7-7485
METOF	032664	7-7486	#7-7489
ME100	= ******	7-4729	
ME101	= ******	7-4742	
ME102	= ******	7-4754	
ME103	= ******	7-4766	
ME104	= ******	7-4778	
ME105	= ******	7-4790	
ME106	= ******	7-4802	
ME107	= ******	7-4814	
ME110	= ******	7-4826	
ME111	= ******	7-4838	
ME112	= ******	7-4850	
ME113	= ******	7-4917	
ME113A	= ******	7-4983	
ME114	= ******	7-5059	
ME115	= ******	7-5376	
ME116	= ******	7-5457	
ME117	= ******	7-5557	
ME120	= ******	7-5601	
ME121	= ******	7-5648	
ME122	= ******	7-5740	
ME123	= ******	7-5818	
ME124	= ******	7-5885	
ME125	= ******	7-5957	
ME126	= ******	7-6123	
ME127	= ******	7-6388	
ME130	= ******	7-6554	
MFA	051640	#7-10642	
MFACU	051636	#7-10634	
MFSRCM	060150	#7-12215	
MIALL	033300	#7-7614	
MIALLA	033324	#7-7622	7-7628
MIALLB	033326	7-7619	#7-7624
MIALLD	033336	7-7625	#7-7628
MIALLF	033346	7-7629	#7-7632
MIL	033100	#7-7556	

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES		
MILA	033124	#7-7564	7-7570	
MILAO	030672	#7-7037		
MILB	033126	7-7561	#7-7566	
MILD	033136	7-7567	#7-7570	
MILF	033146	7-7571	#7-7574	
MILLBO	030772	#7-7063		
MILLO	030204	#7-6914		
MILLOA	030250	7-6920	#7-6925	
MILLOB	030256	7-6922	#7-6928	
MILO	033160	#7-7579		
MILOA	033212	#7-7589		
MILOB	033214	7-7584	#7-7591	
MILOC	033226	7-7592	#7-7595	
MILOD	033252	#7-7600		
MILOE	033254	7-7596	#7-7602	
MILOF	033266	7-7603	#7-7606	
MIOT	032226	#7-7371		
MIOTA	032250	#7-7378	7-7384	
MIOTB	032252	7-7376	#7-7380	
MIOTD	032262	7-7381	#7-7384	
MIOTF	032272	7-7385	#7-7388	
MIOTO	030302	#7-6939		
MITO	032304	#7-7393		
MITOA	032334	#7-7402		
MITOB	032336	7-7398	#7-7404	
MITOC	032350	7-7405	#7-7408	
MITOD	032374	#7-7413		
MITOE	032376	7-7409	#7-7415	
MITOF	032410	7-7416	#7-7419	
MJ	012704	#7-3403		
MJP	013536	#7-3515		
MJP17	013546	#7-3521		
MJP27	013570	#7-3531		
MJP27A	013602	7-3532	#7-3535	
MJP37	013654	7-3536	#7-3553	
MJP37A	013666	7-3554	#7-3557	
MJP67	013612	#7-3537	7-3558	
MJP67A	013624	7-3538	#7-3541	
MJP67B	013640	7-3543	#7-3547	
MJP77	013642	7-3547	#7-3549	
MJP77E	013676	7-3550	#7-3561	
MJRA	014454	#7-3715		
MJR27	014510	#7-3725	7-3731	
MJR27A	014544	7-3732	#7-3735	
MJR27B	014564	#7-3739	7-3764	
MJR37	014646	7-3738	#7-3758	
MJR37A	014702	7-3765	#7-3768	
MJR6A	014642	#7-3755	7-3779	
MJR6B	014644	7-3754	#7-3756	
MJR67	014566	#7-3741	7-3771	
MJR67A	014622	7-3748	#7-3751	
MJR77	014722	7-3747	7-3756	#7-3773

SYMBOL	VALUE	REFERENCES		
MJR77A	014756	7-3780	#7-3783	
MJSI	033360	#7-7639		
MJSIA	033412	#7-7649		
MJSIB	033414	7-7644	#7-7651	
MJSIC	033426	7-7652	#7-7655	
MJSID	033452	#7-7660		
MJSIE	033454	7-7656	#7-7662	
MJSIF	033466	7-7663	#7-7666	
MJSR	013676	#7-3566		
MJSRA	014014	7-3592	#7-3596	7-3646
MJSRB	014160	7-3632	#7-3635	7-3686
MJSRC	014326	7-3673	#7-3675	
MJSR1	014016	7-3573	#7-3598	
MJSR1A	014030	7-3599	#7-3602	
MJSR1B	014054	7-3607	#7-3610	
MJSR2	013734	#7-3578	7-3606	7-3613
MJSR2A	013746	7-3579	#7-3582	
MJSR2B	013772	7-3587	#7-3590	
MJSR3	014162	7-3596	#7-3638	
MJSR3A	014174	7-3639	#7-3642	
MJSR3B	014220	7-3647	#7-3650	
MJSR4	014076	7-3586	#7-3617	7-3653
MJSR4A	014110	7-3618	#7-3621	
MJSR4B	014134	7-3626	#7-3629	
MJSR5	014330	7-3635	#7-3678	7-3705
MJSR5A	014342	7-3679	#7-3682	
MJSR5B	014366	7-3687	#7-3690	
MJSR6	014244	7-3625	#7-3658	7-3693
MJSR6A	014256	7-3659	#7-3662	
MJSR6B	014302	7-3667	#7-3670	
MJSR7	014412	7-3666	7-3675	#7-3697
MJSR7A	014424	7-3698	#7-3701	
MJSR7E	014450	7-3706	#7-3709	
MJU1	012756	7-3407	#7-3421	
MJU1A	012770	7-3422	#7-3425	
MJU2	012720	#7-3409	7-3413	7-3426
MJU2A	012732	7-3410	#7-3413	
MJU2B	012742	7-3414	#7-3417	
MJU3	013002	7-3420	#7-3428	
MJU3A	013014	7-3429	#7-3432	
MJU3B	013024	7-3433	#7-3436	
MJU4	013076	7-3437	#7-3452	7-3453
MJU4A	013110	7-3454	#7-3457	
MJU4B	013122	7-3458	#7-3461	
MJU5	013036	#7-3439	7-3465	
MJU5A	013050	7-3441	#7-3444	
MJU5B	013062	7-3445	#7-3448	
MJU6	013136	7-3449	#7-3467	
MJU6A	013150	7-3468	#7-3471	
MJU7	013170	7-3474	#7-3476	
MJU7E	013202	7-3477	#7-3480	
MJ2	012754	7-3418	#7-3420	7-3432

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
MJ5	013134	7-3440	7-3462	#7-3465						
MJ7	013164	7-3472	#7-3474							
MLCD	076242	#7-15511								
MLCF	075424	#7-15324								
MLDC	063642	#7-13222								
MLDC2	075324	#7-15281								
MLDDM2	060220	#7-12250								
MLDDM3	060306	#7-12289								
MLDDM4	060422	#7-12334								
MLDDM5	060526	#7-12377								
MLDDM6	060616	#7-12414								
MLDDM7	060702	#7-12447								
MLDM27	060770	#7-12484								
MLDSUB	070376	7-14051	7-14059	7-14067	7-14075	7-14083	7-14091	7-14099	7-14107	7-14116
		7-14124	#7-14147							
MLFSUB	067400	7-13837	7-13845	7-13853	7-13861	7-13869	7-13877	7-13885	7-13893	7-13901
		7-13909	7-13917	7-13925	7-13934	7-13943	7-13952	#7-13974		
MLS1	074410	#7-14995								
MLS2	074502	#7-15035								
MLS3	074574	#7-15075								
MLS4	074704	#7-15117								
MLS5	075000	#7-15157								
MLS6	075110	#7-15199								
MLS7	075210	#7-15239								
MLXP	076644	#7-15634								
MMARK	017534	#7-4673								
MMODD	071650	#7-14444								
MMODF	070610	#7-14219								
MMRL5	012436	#7-3230								
MMR0	= 177572	#7-969	*49-20274	*49-20305	*52-20622	*52-20651				
MMR1	= 177574	#7-970								
MMR2	= 177576	#7-971								
MMR3	= 172516	#7-972	*24-18024	*24-18092	*27-18497	*27-18531	*49-20275	*49-20306		
MMTS10	= ******	7-9505								
MMTS11	= ******	7-9589								
MMTS12	= ******	7-9640								
MMTS13	= ******	7-9719								
MMTS14	= ******	7-9951								
MMTS15	= ******	7-10202								
MMTS16	= ******	7-10301								
MMTS6A	= ******	7-8682								
MMTS6B	= ******	7-8786								
MMTS6C	= ******	7-8915								
MMTS6D	= ******	7-9029								
MMU	135066	7-8694	7-8796	7-8925	7-9039	7-9153	7-9171	7-9260	7-9302	7-9513
		7-9525	7-9536	7-9550	7-9614	7-9646	7-9733	7-9982	7-10176	7-10207
		7-10307	#62-21542							
MMULD	067612	#7-14046								
MMULF	066604	#7-13832								
MMUTRP	135240	7-1155	#62-21588							
MMUTST	= ******	7-8080	62-21535							
MMUTS0	= ******	7-3819								

J5

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES		
MMUTS1	= ******	7-8087		
MMUTS2	= ******	7-8107		
MMUTS3	= ******	7-8363		
MMUTS4	= ******	7-8438		
MMUTS5	= ******	7-8626		
MMUTS6	= ******	7-8659		
MMUTS7	= ******	7-9143		
MMUTS8	= ******	7-9250		
MMUTS9	= ******	7-9293		
MMVCC	015530	#7-3952		
MMVEC	= 000250	#5-441		
MM11	010664	#7-2674		
MM22	011232	#7-2822		
MNCCCC	016260	#7-4202		
MNGOP	062106	#7-12813		
MNNRM1	061062	#7-12522		
MNNRM2	061302	#7-12599		
MNNRM3	061434	#7-12651		
MNNRM4	061556	#7-12700		
MNRM	062740	#7-13012		
MODD	= ******	7-14435		
MODE1	046014	7-9742	#7-9829	
MODE2	047374	7-10006	#7-10118	
MODF	= ******	7-14210		
MODGAR	071640	7-14363	#7-14431	7-14599
MRLB1	012164	#7-3121		
MRLCC	016532	#7-4313		
MRLO	012112	#7-3094		
MRL2	012244	#7-3151		
MRL3	012322	#7-3181		
MRL4	012376	#7-3208		
MRL5	= ******	7-3229		
MRL6	012500	#7-3252		
MRL7	012536	#7-3272		
MRRB1	012656	#7-3384		
MRRCC	016612	#7-4348		
MRRO	012634	#7-3368		
MRT	031504	#7-7212		
MRTA	031526	#7-7219	7-7225	
MRTB	031530	7-7217	#7-7221	
MRTE	031540	7-7222	#7-7225	
MRTF	031550	7-7226	#7-7229	
MRTO	031562	#7-7237		
MRTOA	031612	#7-7246		
MRTOB	031614	7-7242	#7-7248	
MRTOC	031626	7-7249	#7-7252	
MRTOD	031652	#7-7258		
MRTOE	031654	7-7253	#7-7260	
MRTOF	031666	7-7261	#7-7264	
MRTS	014762	#7-3790		
MSB	062616	#7-12963		
MSBCC	016376	#7-4251		

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
MSBCCC	016446	#7-4278
MSCD	100234	#7-15893
MSCF	101270	#7-16101
MSDF	073610	#7-14770
MSER	= 177744	#7-959 7-16288 7-16291 9-16499 9-16557 11-16784 *15-17119 15-17120 15-17135 *15-17138 15-17139 15-17151 *16-17217 *16-17227 *17-17307 17-17311 17-17318 *18-17393 18-17406 19-17455 *19-17461 *20-17579 20-17584 *20-17598 20-17604 *20-17609 20-17615 *20-17626 20-17632 *20-17636 20-17642 *20-17647 49-20390 *49-20396 50-20438 *50-20448 51-20501 *51-20502 *51-20509 58-21192 58-21196 58-21197 62-21404 62-21407 62-21410 *62-21413
MSFD	073310	#7-14678
MSFDI	074242	#7-14903
MSOB	017460	#7-4644
MSPAA	007624	#7-2258
MSPAU	027556	#7-6795
MSPB	005716	#7-1544
MSPBB	007710	#7-2294
MSPC	005742	#7-1564
MSPD	005772	#7-1585
MSPEO	006030	#7-1610
MSPF	006072	#7-1637
MSPG	006134	#7-1663
MSPH	006172	#7-1688
MSPI	006236	#7-1716
MSPJ	006276	#7-1742
MSPL	006416	#7-1794
MSPM	006464	#7-1823
MSPN	006542	#7-1857
MSPO	006610	#7-1886
MSPP	006714	#7-1933
MSPQ	007012	#7-1970
MSPR	007120	#7-2013
MSPS	007206	#7-2046
MSPT	007254	#7-2073
MSPU	007272	#7-2089
MSPV	007372	#7-2141
MSPVO	007336	#7-2117
MSPX	007436	#7-2169
MSPY	007506	#7-2199
MSPZ	007554	#7-2228
MSPO	005674	#7-1525
MSTB3	007760	#7-2324
MSTO	031062	#7-7087
MSTOE	031136	7-7091 #7-7103
MSTOEE	031144	7-7102 #7-7106
MST4	010032	#7-2355
MST4B	010070	#7-2380
MST5	010130	#7-2405
MST5B	010162	#7-2428
MST6	010224	#7-2455
MST7	010272	#7-2482
MSW37	012600	#7-3350

L5

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES			
MSXP	101432	#7-16145			
MSXT	017262	#7-4564			
MSXTCC	017120	#7-4489			
MS11	011032	#7-2739			
MS22	011304	#7-2851			
MS33	011534	#7-2956			
MS77	012024	#7-3068			
MTP	031700	#7-7270			
MTPA	032102	#7-7331			
MTPAA	032156	7-7339	7-7340	7-7346	#7-7351
MTPAE	032206	7-7356	#7-7360		
MTPAH	032154	7-7338	#7-7348		
MTPAL	032144	#7-7342	*7-7357	*7-7358	7-7359
MTPB	031732	7-7275	#7-7281		
MTPF	031752	7-7286	#7-7289		
MTPO	031764	#7-7297			
MTPOA	032014	#7-7306			
MTPOB	032016	7-7302	#7-7308		
MTPOC	032030	7-7309	#7-7312		
MTPOD	032054	#7-7317			
MTPOE	032056	7-7313	#7-7319		
MTPOF	032070	7-7320	#7-7323		
MTPQ	031742	7-7282	#7-7285		
MTPR	031730	#7-7279	7-7285		
MTRPO	030476	#7-6987			
MTRY	027720	#7-6841			
MTRYA	027774	7-6848	#7-6857		
MTRYB	030014	7-6861	#7-6865		
MTRYM	030046	#7-6877			
MTSO	015530	7-3888	#7-3949		
MTT	031156	#7-7113			
MTTA	031212	7-7119	#7-7124	7-7130	
MTTB	031214	7-7120	#7-7126		
MTTD	031224	7-7127	#7-7130		
MTTE	031234	7-7131	#7-7134		
MTTR	031340	#7-7171			
MTTRA	031404	7-7177	#7-7184		
MTTRB	031406	7-7178	#7-7186		
MTTRC	031420	7-7187	#7-7190		
MTTRD	031456	7-7192	#7-7199		
MTTRE	031460	7-7193	#7-7201		
MTTRF	031472	7-7202	#7-7205		
MTTS	031246	#7-7142			
MTTSA	031302	7-7148	#7-7153		
MTTSB	031306	7-7149	#7-7157		
MTTSD	031316	7-7158	#7-7161		
MTTSE	031326	7-7162	#7-7165		
MTTSQ	031304	#7-7155	7-7161		
MULD	- ******	7-14037			
MULF	- ******	7-13823			
MUVAD	063050	#7-13055			
MVCC	- ******	7-3951			

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
MXDF1	061666	#7-12742
MXOR	017400	#7-4614
MXRCC	017170	#7-4518
M11	- ******	7-2673
M2	004752	#7-1240
M22	- ******	7-2821
M3	004766	#7-1256
M4	005006	#7-1271
M5	005026	#7-1286
M6	005046	#7-1297
NCCCC	- ******	7-4201
NEWADD	003026	#7-994
NEWDAT	111474	*26-18294 26-18327 26-18334 *26-18345 26-18347 *26-18353 *26-18355 #26-18392 *27-18552 27-18578 27-18593 *27-18597 *27-18602 *27-18605
NGOP	- ******	7-12804
NNRM1	- ******	7-12514
NNRM2	- ******	7-12590
NNRM3	- ******	7-12642
NNRM4	- ******	7-12691
NOABRT	136464	64-21666 64-21666 #64-21666
NOTOK	135316	62-21596 #62-21602
NRM	- ******	7-13003
NULL	- 000000	#7-974 41-19697 44-19881 45-19978 47-20152
NXMFIN	047616	7-10184 7-10194 #7-10196
NXMPAR	106670	21-17742 #21-17756
NXMTRP	047224	7-9966 #7-10069
NXMTST	- ******	7-10162
ODDADR	- ******	7-7698
ODDXX	033600	#7-7699
OK	135274	62-21595 #62-21597
OKAY7	043036	7-9219 #7-9222
OKAY7A	043046	7-9223 #7-9226
OKA7	043024	7-9215 #7-9218
OK1	135322	62-21601 #62-21603
OK7	043006	7-9210 #7-9213
ONOQ22	134630	62-21474 #62-21478
PAR	135210	62-21547 62-21548 62-21551 62-21552 62-21554 62-21555 #62-21573
PARAD1	045644	7-9735 #7-9777
PARAD2	047256	7-10002 #7-10079
PARVA1	045676	7-9736 #7-9790
PARVA2	047310	7-10001 #7-10092
PARVA3	047426	7-9986 7-10008 #7-10131
PAR1	135212	#62-21574 62-21577
PCR	- 177522	#7-960 *28-18669 *28-18676 28-18677 28-18679 *28-18688 28-18689 28-18691 *28-18701 28-18702 *30-18844 30-18855 *30-18867 *30-18868 30-18886 *30-18890 *30-18891 *30-18900 *31-18931 *31-18948 *32-19026 *32-19039 32-19050 *32-19058 32-19059 *32-19070 58-21202
PDR	135166	62-21546 62-21550 62-21553 #62-21564
PDR1	135170	#62-21565 62-21568
PHY1	045762	7-9740 #7-9816
PIR	- 177772	#7-961
PIRQ	- 177772	#5-442 7-7799 *7-7818 7-7819 *7-7823 7-7824 *7-7859 *7-7863 7-7865 *7-7901 7-7912 *7-7913 *7-7934 *7-7940 *7-7944 *7-7947 *7-7969 *7-7978

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		7-7981	7-7998	*7-7999	*7-8047	*7-8052	7-8062	*7-8071	*54-20814	*54-20829
PIRQNX	034202	7-7797	#7-7803							
PIRQT	123252	54-20808	#54-20831							
PIRQVE	= 000240	#5-442	7-7808	7-7809	*7-7810	*7-7811	*7-7897	*7-7898	*7-7938	*7-7939
		*7-7970	*7-7971	*7-8049	*7-8050	*7-8074	*7-8075	*54-20805	*54-20806	
PIRRTN	034502	7-7897	#7-7912							
PIRTBL	034304	7-7815	#7-7834							
PIRTEX	034204	7-7801	#7-7804							
PIRTST	= ******	7-7792								
PIRXXX	034146	#7-7794								
PIR1	034204	#7-7806								
PIR2	034324	#7-7855								
PIR2EX	034432	7-7869	#7-7884							
PIR3	034432	#7-7893								
PIR3EX	034526	7-7910	#7-7920							
PIR4	034526	#7-7931								
PIR5	034604	#7-7966								
PIR5EX	035014	7-7993	#7-8027							
PIR6	035014	#7-8044								
PIR6EX	035116	7-8059	#7-8074							
PITBL1	034410	7-7862	#7-7879							
PITBL2	035000	7-7972	#7-8020							
PI1	034454	#7-7900	7-7909							
PI2	034466	#7-7906	7-7916							
PI3	034470	#7-7907	7-7917							
PLF0	043654	7-9303	#7-9380							
PLF1	044046	7-9314	7-9325	#7-9444						
POLY	= 120001	#7-973								
PROCNT	023106	#7-5622	7-5635							
PR0	= 000000	#5-442								
PR1	= 000040	#5-442								
PR2	= 000100	#5-442								
PR3	= 000140	#5-442								
PR4	= 000200	#5-442								
PR5	= 000240	#5-442								
PR6	= 000300	#5-442								
PR7	= 000340	#5-442	7-7811	7-7971	7-8050					
PS	= 177776	#5-442	5-442	*7-5139	7-5140	*7-5151	7-5152	*7-5180	*7-5187	*7-5194
		*7-5372	*7-5619	*7-7179	*7-7194	*7-7243	*7-7254	*7-7276	*7-7303	*7-7314
		*7-7399	*7-7410	*7-7463	*7-7477	*7-7528	*7-7539	*7-7585	*7-7597	*7-7645
		*7-7657	*7-7679	*7-7738	7-7739	7-7743	7-7747	*7-7976	7-7980	*7-10182
PSW	= 177776	#5-442	*7-7769	*7-7935						
PSWBTS	005576	#7-1492								
PWRVEC	= 000024	#5-442	*7-1149	*7-1149	55-20856	*55-20857	*55-20858	*55-20867	*64-21674	*64-21674
		*64-21674	*64-21674	*64-21674	*64-21674					
Q22EN	003002	#7-948	53-20727	53-20756	54-20807	56-20902				
Q22INT	134664	53-20738	53-20767	54-20815	56-20911	56-20915	#62-21485			
Q22SIZ	134404	7-1191	#62-21423							
RAMPAR	134326	7-1153	#62-21403							
RBUF	= 177562	#7-962	42-19751	42-19764	44-19906	45-19958	45-19992	46-20056	46-20062	46-20083
		47-20136	47-20155	48-20205	58-21207					
RCSR	= 177560	#7-963	40-19657	42-19747	42-19757	42-19760	42-19770	*44-19843	44-19844	44-19851

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
		•44-19882 •44-19901 44-19903 45-19955 •45-19979 •45-19996 46-20053 46-20060 46-20081
		47-20129 47-20153 •48-20189
RDCHR	= 104410	64-21671 #64-21673
RDLIN	= 104411	64-21672 #64-21673
RDOCT	= 104412	#64-21673
RECDAT	111464	•9-16544 9-16545 •9-16549 9-16550 10-16624 •10-16643 •11-16754 11-16755 •11-16760
		11-16761 •11-16770 11-16771 •12-16850 12-16851 •12-16860 12-16861 •12-16866 12-16867
		•13-16938 13-16939 •13-16954 13-16955 •14-17018 14-17019 •14-17023 14-17024 •14-17028
		14-17029 •14-17033 14-17034 •15-17146 15-17147 •16-17222 16-17223 •22-17804 22-17805
		22-17808 •22-17815 22-17816 22-17819 •22-17824 22-17825 22-17829 •23-17916 23-17921
		•23-17941 23-17946 •24-18061 24-18062 •26-18269 •26-18323 26-18324 •26-18333 26-18334
		26-18380 #26-18388 •27-18574 27-18575 •27-18580 27-18581 27-18583 •27-18587 27-18588
		•49-20287 49-20289 •52-20561 52-20563 •52-20581 52-20583 •52-20605 52-20607 •52-20638
		52-20640 •52-20679 52-20681 58-21194 58-21199
RECDST	003142	#7-1014 7-12253 7-12292 7-12314 7-12337 7-12380 7-12417 7-12450 7-12487
		•7-12526 •7-12527 7-12537 7-12602 7-12654 7-12707 7-12748 7-12822 7-12844
		7-12874 7-12901 7-12919 7-12937 7-12969 7-13023 7-13036 7-13065 7-13089
		7-13116 7-13140 7-13167 7-13191 7-13232 7-13256 7-13288 7-13308 7-13391
		7-13630 7-13797 7-14011 7-14184 7-14397 7-14633 7-14731 7-14837 7-14939
		7-15303 7-15467 7-15588 7-15817 7-16027 7-16201 7-16219
RECFEC	003122	#7-1012 7-12220 7-14884 7-15873 7-16081
RECST	003132	#7-1013 7-15000 7-15008 7-15012 7-15040 7-15048 7-15052 7-15082 7-15094
		7-15122 7-15130 7-15134 7-15164 7-15176
REDZON	= ******	7-7761
RESVEC	= 000010	#5-442
RET1	015246	#7-3872 7-3904 7-3919 7-3942
RET2	015334	#7-3894 7-3924 7-3937
RET3	015412	#7-3914 7-3932
RITEDA	111472	•26-18293 26-18330 26-18332 26-18339 •26-18344 •26-18349 •26-18350 26-18353 26-18355
		#26-18391 •27-18553 •27-18596 27-18597 •27-18601 •27-18604 27-18605
RLB1	= ******	7-3120
RLCC	= ******	7-4312
RLO	= ******	7-3093
RL2	= ******	7-3150
RL3	= ******	7-3180
RL4	= ******	7-3207
RL6	= ******	7-3251
RL7	= ******	7-3271
ROM	= ******	7-3483
RRB1	= ******	7-3383
RRCC	= ******	7-4347
RRO	= ******	7-3367
RT	= ******	7-7211
RTO	= ******	7-7236
RTS	= ******	7-3789
RTSE	015034	7-3813 #7-3816
RTS1	015002	7-3795 #7-3799
RTS6	015012	7-3800 #7-3806
RXXX	034020	#7-7762
R6	=%000006	#5-442 •7-1149 •7-1149 7-1149 •7-1467 7-1468 •7-1472 7-1473 •7-1477
		7-1478 •7-1482 7-1483 •7-1487 7-3569 7-3570 7-3582 7-3584 •7-3591
		7-3602 7-3604 •7-3611 7-3621 7-3623 •7-3630 7-3642 7-3644 •7-3651

C6

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		7-3662	7-3664	*7-3671	7-3682	7-3684	*7-3691	7-3701	7-3703	*7-3710
		7-3719	7-3720	7-3727	7-3729	*7-3737	7-3743	7-3745	*7-3752	7-3760
		7-3762	*7-3769	7-3775	7-3777	*7-3785	*7-3793	*7-3794	7-3806	*7-3808
		*7-3809	7-3812	*7-3816	*7-3823	7-3824	*7-3828	7-3829	*7-3833	7-3834
		*7-3838	7-3839	*7-3843	*7-3845	7-3846	*7-3850	7-3851	*7-3855	7-3856
		*7-3860	7-3861	*7-3865	*7-3867	7-3873	7-3878	7-3883	7-4718	7-5069
		7-5076	7-5086	7-5401	*7-5403	7-5408	*7-5412	*7-5470	*7-5620	*7-5757
		*7-5996	7-6009	*7-6011	*7-6127	*7-6133	*7-6134	*7-6139	7-6144	*7-6146
		*7-6151	*7-6798	*7-6799	7-6800	*7-6804	*7-6805	*7-6806	7-6807	*7-6811
		*7-6812	7-6813	*7-6817	*7-6818	*7-6819	7-6820	*7-6824	*7-6825	7-6826
		*7-6830	*7-6831	7-6832	*7-6836	*7-6845	*7-6853	*7-6854	7-6860	*7-6871
		*7-6881	*7-6885	*7-6886	*7-6892	*7-6894	*7-6895	*7-6900	*7-6902	*7-6903
		*7-6908	*7-6918	*7-6925	*7-6932	*7-6943	*7-6951	*7-6956	*7-6968	*7-6976
		*7-6983	*7-6991	*7-6999	*7-7006	*7-7016	*7-7024	*7-7031	*7-7041	*7-7050
		*7-7057	*7-7066	*7-7075	*7-7081	*7-7092	*7-7093	*7-7094	*7-7095	*7-7096
		*7-7097	*7-7098	*7-7099	*7-7100	*7-7101	*7-7103	*7-7107	*7-7116	*7-7118
		*7-7119	7-7126	7-7130	*7-7137	*7-7145	*7-7147	*7-7148	7-7157	7-7161
		*7-7167	*7-7174	*7-7176	*7-7177	*7-7190	*7-7191	*7-7192	*7-7207	*7-7215
		7-7221	7-7225	*7-7231	*7-7240	*7-7252	*7-7266	*7-7273	7-7281	7-7285
		*7-7291	*7-7300	*7-7312	*7-7325	*7-7335	*7-7354	*7-7365	*7-7374	7-7380
		7-7384	*7-7389	*7-7396	*7-7408	*7-7421	*7-7429	7-7440	7-7444	*7-7450
		*7-7458	*7-7475	*7-7492	*7-7501	7-7507	7-7511	*7-7516	*7-7525	*7-7537
		*7-7550	*7-7559	7-7566	7-7570	*7-7575	*7-7582	*7-7595	*7-7608	*7-7617
		7-7624	7-7628	*7-7633	*7-7642	*7-7655	*7-7668	*7-7767	7-8689	7-8692
		7-8709	7-8714	7-8792	7-8794	7-8811	7-8816	7-8921	7-8923	7-8941
		7-8946	7-9035	7-9037	7-9055	7-9060	*7-9947	*7-10333	*7-10335	7-10376
		*7-10400	7-10402	*7-10426	*7-10428	7-10441	*7-10457	7-10459	*7-14971	
R7	=#000007	#5-442	*7-2079	7-3519	7-9207	7-9267	7-9339	7-9519	7-9530	7-9541
		7-9554	7-9657	7-9664	7-9671	7-9678	7-9685	7-9692	7-10410	*7-10549
		*7-10556	*7-10563	*7-10570	*7-10578	*7-10586	*7-10615	7-12221	*7-12270	*7-12315
		*7-12358	*7-12397	*7-12430	*7-12467	*7-12506	*7-12539	*7-12554	*7-12567	*7-12581
		*7-12618	*7-12633	*7-12667	*7-12682	*7-12712	*7-12724	*7-12759	*7-12769	*7-12783
		*7-12794	*7-12829	*7-12857	*7-12887	*7-12908	*7-12926	*7-12944	*7-12979	*7-12994
		*7-13026	*7-13038	*7-13072	*7-13102	*7-13123	*7-13153	*7-13174	*7-13204	*7-13239
		*7-13263	*7-13295	*7-13315	*7-13338	*7-13345	*7-13352	*7-13358	*7-13402	*7-13427
		*7-13437	*7-13446	*7-13454	*7-13463	*7-13471	*7-13479	*7-13487	*7-13495	*7-13503
		*7-13511	*7-13519	*7-13527	*7-13536	*7-13544	*7-13553	*7-13562	*7-13571	*7-13638
		*7-13670	*7-13679	*7-13688	*7-13696	*7-13704	*7-13712	*7-13720	*7-13728	*7-13737
		*7-13805	*7-13837	*7-13845	*7-13853	*7-13861	*7-13869	*7-13877	*7-13885	*7-13893
		*7-13901	*7-13909	*7-13917	*7-13925	*7-13934	*7-13943	*7-13952	*7-14019	*7-14051
		*7-14059	*7-14067	*7-14075	*7-14083	*7-14091	*7-14099	*7-14107	*7-14116	*7-14124
		*7-14192	*7-14224	*7-14233	*7-14242	*7-14251	*7-14260	*7-14270	*7-14279	*7-14288
		*7-14297	*7-14306	*7-14315	*7-14324	*7-14333	*7-14406	*7-14415	*7-14449	*7-14458
		*7-14467	*7-14476	*7-14485	*7-14494	*7-14503	*7-14512	*7-14521	*7-14530	*7-14540
		*7-14549	*7-14559	*7-14569	*7-14642	*7-14651	*7-14682	*7-14688	*7-14694	*7-14700
		*7-14706	*7-14747	*7-14775	*7-14782	*7-14789	*7-14796	*7-14803	*7-14810	*7-14875
		*7-14943	*7-15288	*7-15305	*7-15328	*7-15334	*7-15340	*7-15346	*7-15352	*7-15358
		*7-15364	*7-15370	*7-15376	*7-15382	*7-15388	*7-15394	*7-15400	*7-15406	*7-15412
		*7-15418	*7-15424	*7-15430	*7-15436	*7-15442	*7-15486	*7-15515	*7-15521	*7-15527
		*7-15533	*7-15539	*7-15545	*7-15551	*7-15557	*7-15563	*7-15607	*7-15639	*7-15647
		*7-15655	*7-15663	*7-15671	*7-15679	*7-15688	*7-15697	*7-15706	*7-15715	*7-15724
		*7-15732	*7-15740	*7-15749	*7-15758	*7-15766	*7-15775	*7-15783	*7-15792	*7-15864

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		#7-15898	#7-15905	#7-15912	#7-15919	#7-15926	#7-15933	#7-15940	#7-15947	#7-15954
		#7-15961	#7-15968	#7-15975	#7-15982	#7-15989	#7-15996	#7-16003	#7-16072	#7-16106
		#7-16113	#7-16120	#7-16127	#7-16149	#7-16155	#7-16161	#7-16167	#7-16173	#7-16179
		#63-21622	#63-21623	#63-21658						
SAVBR	002730	#7-943	#29-18743	29-18792	#30-18837	#30-18902	#31-18923	31-18949	#32-19022	32-19077
		#35-19315	35-19337	#36-19386	36-19430	#37-19469	#38-19543	38-19583	#46-20046	46-20077
		#50-20431	50-20450	#57-21004	57-21051					
SAVMR0	003042	#7-1000	#7-9150	#7-9181	#7-9213	7-9214	#7-9226	#7-9237	7-9238	#7-9257
		#7-9273	7-9274	#7-9298	#7-9353	7-9354	#7-9367	#7-9510	7-9570	#7-9582
		7-10420	#62-21603							
SAVMR1	003044	#7-1001	#7-9151	#7-9182	7-9218	#7-9227	7-9242	#7-9258	7-9278	#7-9299
		7-9358	#7-9368	#7-9511	7-9574	#7-9583	#7-9645	7-9705	#7-9713	#62-21604
SAVMR2	003046	#7-1002	#7-9152	#7-9183	7-9222	#7-9228	#7-9259	7-9282	#7-9300	7-9362
		#7-9369	#7-9512	7-9578	#7-9584	7-9709	#7-9714	7-10416	#62-21605	
SAVPCR	002726	#7-942								
SAVPOS	003156	#7-1021								
SAVSUP	003036	#7-998	#7-8692	7-8714	#7-8794	7-8816	#7-8923	7-8946	#7-9037	7-9060
SAVSWR	003050	#7-1003	#32-19005	#32-19020	33-19126					
SAVUSE	003040	#7-999	#7-8689	7-8709	#7-8792	7-8811	#7-8921	7-8941	#7-9035	7-9055
SAV30	004112	7-1148	#7-1148	#7-1148	64-21666	64-21666				
SAV32	004114	#7-1148	#7-1148	64-21666	64-21666					
SB	= ******	7-12954								
SBCC	= ******	7-4250								
SBCCC	= ******	7-4277								
SCD	= ******	7-15884								
SCDSUB	101054	7-15898	7-15905	7-15912	7-15919	7-15926	7-15933	7-15940	7-15947	7-15954
		7-15961	7-15968	7-15975	7-15982	7-15989	7-15996	7-16003	#7-16025	7-16106
		7-16113	7-16120	7-16127						
SCF	= ******	7-16092								
SCOPE	= 000004	#5-442	7-1195	7-16270	8-16328	9-16422	9-16528	10-16597	11-16739	12-16833
		13-16918	14-17005	15-17110	16-17207	17-17276	18-17364	19-17430	20-17565	21-17721
		22-17791	23-17889	24-18011	25-18109	26-18230	27-18483	28-18668	29-18742	30-18836
		31-18922	32-19004	33-19125	34-19210	35-19307	36-19378	37-19459	38-19525	39-19604
		40-19645	41-19685	42-19729	43-19790	44-19832	45-19941	46-20034	47-20120	48-20175
		49-20256	49-20343	50-20415	51-20468	52-20535	53-20716	54-20798	55-20851	56-20889
		57-20961								
SDF	= ******	7-14761								
SDFSUB	074040	7-14775	7-14782	7-14789	7-14796	7-14803	7-14810	#7-14832		
SDPAR0	= 172260	#5-441								
SDPAR1	= 172262	#5-441								
SDPAR2	= 172264	#5-441								
SDPAR3	= 172266	#5-441								
SDPAR4	= 172270	#5-441								
SDPAR5	= 172272	#5-441								
SDPAR6	= 172274	#5-441								
SDPAR7	= 172276	#5-441								
SDPDR0	= 172220	#5-441								
SDPDR1	= 172222	#5-441								
SDPDR2	= 172224	#5-441								
SDPDR3	= 172226	#5-441								
SDPDR4	= 172230	#5-441								
SDPDR5	= 172232	#5-441								

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
SDPDR6	= 172234	#5-441
SDPDR7	= 172236	#5-441
SEQ	003072	#7-1006 *7-3406 7-3409 *7-3417 7-3421 *7-3425 7-3428 *7-3436 7-3444 *7-3448 7-3457 *7-3461 7-3467 *7-3471 7-3476 *7-3518 7-3521 *7-3525 7-3531 *7-3535 7-3537 *7-3541 7-3549 7-3553 *7-3557 *7-3572 7-3578 *7-3590 7-3598 *7-3610 7-3617 *7-3629 7-3638 *7-3650 7-3658 *7-3670 7-3678 *7-3690 7-3697 *7-3718 7-3725 *7-3735 7-3741 *7-3751 7-3758 *7-3768 7-3773
SFD	= ******	7-14669
SFDI	= ******	7-14894
SFDSUB	073504	7-14682 7-14688 7-14694 7-14700 7-14706 #7-14726
SIMGOA	002702	#7-929 57-20989 57-21046 *62-21448 *62-21449
SIPAR0	= 172240	#5-441
SIPAR1	= 172242	#5-441
SIPAR2	= 172244	#5-441
SIPAR3	= 172246	#5-441
SIPAR4	= 172250	#5-441
SIPAR5	= 172252	#5-441
SIPAR6	= 172254	#5-441
SIPAR7	= 172256	#5-441
SIPDR0	= 172200	#5-441 *7-10406
SIPDR1	= 172202	#5-441
SIPDR2	= 172204	#5-441
SIPDR3	= 172206	#5-441
SIPDR4	= 172210	#5-441
SIPDR5	= 172212	#5-441
SIPDR6	= 172214	#5-441
SIPDR7	= 172216	#5-441
SIXBIT	112774	28-18670 28-18671 #28-18708
SLEND	120134	40-19650 #47-20161
SLOC00	003012	#7-986 *7-5069 7-5076 7-5086 *7-6847 7-6867 *7-6882 7-6907 *7-6919 7-6930 *7-6944 7-6958 *7-6969 7-6980 *7-6992 7-7003 *7-7017 7-7029 *7-7042 7-7055 *7-7067 7-7080 *7-7090 7-7106 *7-7117 7-7135 *7-7146 7-7166 *7-7175 7-7206 *7-7216 7-7230 *7-7241 7-7265 *7-7274 7-7290 *7-7301 7-7324 *7-7336 7-7362 *7-7375 7-7388 *7-7397 7-7420 *7-7430 7-7449 *7-7459 7-7491 *7-7502 7-7515 *7-7526 7-7549 *7-7560 7-7574 *7-7583 7-7607 *7-7618 7-7632 *7-7643 7-7667 *7-7677 7-7695 *7-7712 7-7757 *7-7765 7-7786 *7-7796 7-7800 *7-7808 7-8074 *7-9657 *7-9664 *7-9671 *7-9678 *7-9685 *7-9692 7-9709 *7-9874 7-9948 *7-9964 7-9978 7-10071 *7-10178 7-10198 *7-16279 7-16299 *9-16429 9-16492 *9-16536 9-16553 *11-16746 11-16779 *15-17117 15-17155 15-17160 *16-17214 16-17226 *17-17284 17-17305 *18-17372 18-17392 *19-17445 19-17473 *20-17573 20-17649 *21-17729 21-17750
SLOC01	003014	#7-987 *7-6921 7-6929 *7-6946 7-6957 *7-6971 7-6981 *7-6994 7-7004 *7-7019 7-7030 *7-7044 7-7054 *7-7069 7-7079 *7-7337 7-7363 *7-7432 7-7448 *7-7461 7-7490 *7-7714 7-7758 *7-7809 7-8075 *7-9875 7-9949 *7-9965 7-9979 7-10072 *17-17285 17-17306 *21-17730 21-17751
SOB	= ******	7-4643
SOPAA	= ******	7-2257
SOPBB	= ******	7-2293
SOPD	= ******	7-1584
SOPE	= ******	7-1609
SOPF	= ******	7-1636

F6

SEQ 0484

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
SOPG	= ******	7-1662								
SOPH	= ******	7-1687								
SOPI	= ******	7-1715								
SOPJ	= ******	7-1741								
SOPL	= ******	7-1793								
SOPM	= ******	7-1822								
SOPN	= ******	7-1856								
SOPO	= ******	7-1885								
SOPP	= ******	7-1932								
SOPQ	= ******	7-1969								
SOPR	= ******	7-2012								
SOPS	= ******	7-2045								
SOPT	= ******	7-2072								
SOPU	= ******	7-2088								
SOPV	= ******	7-2140								
SOPVO	= ******	7-2116								
SOPX	= ******	7-2168								
SOPY	= ******	7-2198								
SOPZ	= ******	7-2227								
SOP1	= ******	7-1543								
SOP2	= ******	7-1563								
SPAU	= ******	7-6794								
SPAU1	027574	7-6801	#7-6804							
SPAU2	027616	7-6808	#7-6811							
SPAU3	027636	7-6814	#7-6817							
SPAU4	027662	7-6821	#7-6824							
SPAU5	027676	7-6827	#7-6830							
SPAU6	027714	7-6833	#7-6836							
SPS	003074	#7-1007	*7-3569	7-3591	7-3611	7-3630	7-3651	7-3671	7-3691	7-3710
		*7-3719	7-3737	7-3752	7-3769	7-3785				
SPSJ	003076	#7-1008	*7-3570	*7-3571	7-3582	7-3602	7-3621	7-3642	7-3662	7-3682
		7-3701	*7-3720	*7-3721	7-3727	7-3743	7-3760	7-3775		
SPT3B	= ******	7-2323								
SPO	= ******	7-1516								
SRO	= 177572	#5-441	7-969	7-5681	7-8631	7-8638	7-8645	7-8652	*7-9880	*7-9901
		*7-9915	*7-9924	*7-9933	*7-9942	*7-9945	*7-9962	*7-9975	*7-10069	*7-10180
		*7-10196	*12-16844	*12-16846	*12-16855	*12-16858	*12-16864	*12-16870	*13-16928	*13-16961
		*21-17733	*21-17748	*23-17899	*23-17952	*24-18023	*24-18090	*26-18244	*26-18279	*27-18496
		*27-18530								
SR1	= 177574	#5-441	7-970							
SR2	= 177576	#5-441	7-971							
SR3	= 172516	#5-441	7-972	*7-9878	*7-9894	*7-9908	*7-9974			
STACK	= 001100	#5-442	7-1149							
START	004024	5-447	5-450	#7-1148						
STBOT	= 001000	#7-1042	7-1487	7-3793	7-3867	*7-3870	7-3883	7-4676	*7-4677	7-4694
		7-4718	7-5996	7-6009	7-6011	7-6134	7-6139	7-6146	7-6151	7-6811
		7-6817	7-6830	7-6836	7-6854	7-6871	7-6886	7-6895	7-6903	7-6908
		7-6925	7-6932	7-6951	7-6956	7-6976	7-6983	7-6999	7-7006	7-7024
		7-7031	7-7050	7-7057	7-7075	7-7081	7-7103	7-7107	7-7116	7-7126
		7-7137	7-7145	7-7157	7-7167	7-7174	7-7186	7-7190	7-7201	7-7207
		7-7215	7-7221	7-7231	7-7240	7-7248	7-7252	7-7260	7-7266	7-7273
		7-7281	7-7291	7-7300	7-7308	7-7312	7-7319	7-7325	7-7335	7-7354

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
		7-7365	7-7374	7-7380	7-7389	7-7396	7-7404	7-7408	7-7415	7-7421
		7-7429	7-7440	7-7450	7-7458	7-7471	7-7475	7-7485	7-7492	7-7501
		7-7507	7-7516	7-7525	7-7533	7-7537	7-7544	7-7550	7-7559	7-7566
		7-7575	7-7582	7-7591	7-7595	7-7602	7-7608	7-7617	7-7624	7-7633
		7-7642	7-7651	7-7655	7-7662	7-7668	7-7785	7-8688	7-8691	7-9890
		7-9947								
STKLMT	= 177774	#5-442								
STMOVI	110642	26-18257	#26-18283							
STMOVT	111764	27-18516	#27-18536							
ST0	= ******	7-7086								
ST4	= ******	7-2354								
ST4B	= ******	7-2379								
ST5	= ******	7-2404								
ST5B	= ******	7-2427								
ST6	= ******	7-2454								
ST7	= ******	7-2481								
SUBT	052126	7-10652	7-10655	7-10657	7-10659	7-10661	#7-10703			
SWR	001140	#6-454	7-1149	*7-1149	7-1149	*7-1149	*7-1149	7-1162	7-9857	26-18238
		27-18491	32-19005	32-19011	32-19020	32-19021	33-19126	64-21665	64-21665	64-21665
		64-21665	64-21665	64-21666	64-21666	64-21666	64-21666	64-21671	64-21671	64-21674
		64-21674								
SWREG	000176	#5-444	7-1149	7-1162	64-21671	64-21671				
SW0	= 000001	#5-442	7-3291							
SW00	= 000001	#5-442	5-442							
SW01	= 000002	#5-442	5-442							
SW02	= 000004	#5-442	5-442							
SW03	= 000010	#5-442	5-442							
SW04	= 000020	#5-442	5-442							
SW05	= 000040	#5-442	5-442							
SW06	= 000100	#5-442	5-442							
SW07	= 000200	#5-442	5-442							
SW08	= 000400	#5-442	5-442							
SW09	= 001000	#5-442	5-442							
SW1	= 000002	#5-442	7-3331							
SW10	= 002000	#5-442								
SW11	= 004000	#5-442								
SW12	= 010000	#5-442								
SW13	= 020000	#5-442								
SW14	= 040000	#5-442								
SW15	= 100000	#5-442								
SW2	= 000004	#5-442								
SW3	= 000010	#5-442								
SW37	= ******	7-3349								
SW4	= 000020	#5-442								
SW5	= 000040	#5-442								
SW6	= 000100	#5-442								
SW7	= 000200	#5-442								
SW8	= 000400	#5-442								
SW9	= 001000	#5-442								
SXP	= ******	7-16136								
SXPSUB	101612	7-16149	7-16155	7-16161	7-16167	7-16173	7-16179	#7-16199		
SXT	= ******	7-4554								

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
SXTCC	= ******	7-4488								
S11	= ******	7-2738								
S22	= ******	7-2850								
S33	= ******	7-2955								
S77	= ******	7-3067								
TAB1	003234	#7-1046	7-12254	7-12264	7-12268	7-12382	7-12396	7-12452	7-12466	7-12656
TAB10	003364	#7-1087	7-12749	7-12782						
TAB11	003374	#7-1091	7-12747							
TAB11A	003404	#7-1095	7-12758							
TAB12	003414	#7-1096	7-12764							
TAB13	003424	#7-1097								
TAB13B	003434	#7-1101	7-12767							
TAB14	003444	#7-1102	7-12775	7-12984						
TAB15	003454	#7-1106	7-12776							
TAB16	003464	#7-1110	7-12788	7-13248						
TAB17	003474	#7-1114	7-12793							
TAB18	003504	#7-1117	7-12681							
TAB2	003244	#7-1050	7-12294	7-12313	7-12418	7-12604				
TAB21	003514	#7-1121	7-12818	7-12835	7-12864	7-12968				
TAB22	003524	#7-1122	7-12828							
TAB23	003534	#7-1123	7-12836	7-12863						
TAB24	003544	#7-1124	7-12856	7-12886						
TAB25	003554	#7-1125	7-12894	7-12914						
TAB26	003564	#7-1126	7-12893	7-12915						
TAB27	003574	#7-1127	7-12907	7-12925						
TAB28	003604	#7-1128	7-12933							
TAB29	003614	#7-1129	7-12932							
TAB29A	003624	#7-1130	7-12943							
TAB3	003254	#7-1054	7-12338	7-12353	7-12357					
TAB30	003634	#7-1132	7-13018							
TAB31	003644	#7-1133	7-13017							
TAB32	003654	#7-1134	7-13031							
TAB33	003664	#7-1135	7-13032							
TAB34	003674	#7-1136	7-13025							
TAB4	003264	#7-1058	7-12617	7-12632						
TAB40	003704	#7-1137	7-13060	7-13061	7-13080	7-13081				
TAB41	003714	#7-1138	7-13112	7-13133	7-13162	7-13183				
TAB42	003724	#7-1139	7-13163	7-13184						
TAB43	003734	#7-1140	7-13203							
TAB45	003744	#7-1141	7-13228	7-13234	7-13247	7-13251	7-13284			
TAB46	003754	#7-1142	7-13238							
TAB47	003764	#7-1143	7-13283	7-14935						
TAB47A	003774	#7-1144								
TAB48	004004	#7-1145	7-13294	7-13304						
TAB49	004014	#7-1146	7-13262							
TAB5	003274	#7-1062	7-12488							
TAB5A	003304	#7-1066	7-13152							
TAB6	003314	#7-1070	7-12339	7-12525	7-12603	7-12655	7-12978	7-12993	7-13071	7-13101
		7-13122	7-13173	7-13229	7-13303	7-13314	7-13399	7-14942		
TAB6A	003324	#7-1074	7-15302							
TAB7	003334	#7-1078	7-12705	7-13111	7-13132					
TAB8	003344	#7-1082	7-12706	7-12717	7-12718					

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
TAB9	003354	#7-1086	7-12711							
TAPABO	105034	16-17215	#16-17232							
TA114	021150	7-5094	#7-5099							
TA116	022624	7-5483	7-5492	#7-5518	7-5533					
TBITVE	= 000014	#5-442								
TB114	021160	7-5100	#7-5103							
TC114	021170	7-5104	#7-5107							
TD114	021200	7-5108	#7-5111							
TEMP	002740	#7-945	49-20352	49-20373	49-20379	49-20387	50-20430	*50-20437	*51-20494	*52-20555
		52-20560	52-20566	52-20580	52-20590	52-20594	52-20602	*52-20654	52-20664	52-20665
		52-20674	52-20678	57-20981	57-20982	57-20994	57-21013	57-21017	57-21022	57-21039
		57-21043	57-21048	62-21503	62-21514					
TE102	017740	#7-4755								
TE103	017762	#7-4767								
TE104	020004	#7-4779								
TE105	020026	#7-4791								
TE106	020050	#7-4803								
TE107	020072	#7-4815								
TE110	020114	#7-4827								
TE111	020136	#7-4839								
TE112	020160	#7-4851								
TE113	020402	#7-4918								
TE113A	020622	#7-4984								
TE114	020774	#7-5060								
TE115	022144	#7-5377								
TE115A	022354	7-5381	7-5398	7-5415	#7-5441					
TE115B	022364	7-5382	7-5399	7-5416	#7-5445					
TE115C	022372	7-5383	7-5400	#7-5448						
TE115D	022400	7-5417	#7-5451							
TE115F	022406	7-5437	#7-5454							
TE116	022410	#7-5458								
TE116A	022664	7-5479	7-5488	#7-5536						
TE116B	022670	7-5463	7-5489	#7-5540						
TE116C	022702	7-5472	7-5480	#7-5545						
TE116D	022712	7-5462	7-5471	7-5481	7-5490	#7-5549				
TE117	022722	#7-5565								
TE117A	023034	7-5568	#7-5595							
TE120	023040	#7-5611								
TE120A	023114	7-5617	#7-5627							
TE121	023170	#7-5649								
TE122	023554	#7-5741								
TE123	024026	#7-5819								
TE124	024246	#7-5886								
TE125	024500	#7-5958								
TE125A	024740	7-5961	7-5993	#7-6031						
TE126	025166	#7-6124								
TE126A	025234	7-6131	#7-6137							
TE126B	025710	7-6226	#7-6265							
TE127	026226	#7-6389								
TE127A	026426	7-6419	#7-6451							
TE130	026670	#7-6555								
TE130A	027146	7-6602	#7-6639							

J6

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
TF114	021210	7-5112	#7-5115							
TG114	021220	7-5116	#7-5119							
THRBIT	113242	29-18748	#29-18795							
TH114	021230	7-5120	#7-5123							
TIMDEL	116116	38-19546	38-19571	#38-19587						
TIMEOU	052740	7-10827	#7-10879							
TIMOUT	003000	#7-947								
TKVEC	= 000060	#5-442								
TM16A	050514	7-10315	#7-10372							
TM16B	050374	7-10313	#7-10344							
TM16C	050424	7-10319	#7-10352							
TM16D	050454	7-10325	#7-10360							
TM16E	050504	7-10306	#7-10368	7-10380	7-10384					
TM16F	050506	#7-10369	7-10407							
TM16A	051100	7-10341	7-10371	7-10436	7-10460	#7-10463				
TOUT	135056	7-1151	7-16278	52-20653	#62-21531					
TP	= ******	7-7269								
TPA	= ******	7-7330								
TPO	= ******	7-7296								
TPVEC	= 000064	#5-442								
TRAPVE	= 000034	#5-442	*7-1149	*7-1149						
TRPFLG	135370	*7-10732	*7-10758	*7-10787	*7-10822	*7-10892	*7-10923	*7-10978	*7-11042	*7-11106
		*7-11139	*7-11161	*7-11202	*7-11249	*7-11307	*7-11359	*7-11414	*7-11467	*7-11525
		*7-11574	*7-11615	*7-11668	*7-11710	*7-11813	*7-11909	*7-12005	*7-12050	*7-12132
		#63-21621								
TRPO	= ******	7-6986								
TRPOA	030542	7-6993	#7-6999							
TRPOB	030550	7-6995	#7-7002							
TRTVEC	= 000014	#5-442								
TRY	= ******	7-6840								
TRYM	= ******	7-6876								
TRYMA	030106	7-6883	#7-6889							
TRYMB	030140	7-6893	#7-6898							
TRYMC	030166	7-6901	#7-6906							
TSD31	057326	#7-12010	7-12019							
TSEND	112612	27-18631	#27-18635							
TSFP1	052252	#7-10730								
TSFP10	053434	#7-11040								
TSFP11	053654	#7-11104								
TSFP12	053750	#7-11137								
TSFP13	054010	#7-11157								
TSFP14	054134	#7-11198								
TSFP15	054266	#7-11245								
TSFP16	054454	#7-11303								
TSFP17	054620	#7-11355								
TSFP2	052336	#7-10756								
TSFP20	055002	#7-11412								
TSFP21	055154	#7-11465								
TSFP22	055344	#7-11523								
TSFP23	055510	#7-11572								
TSFP24	055630	#7-11613								
TSFP25	056004	#7-11666								

K6

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES				
TSFP26	056122	#7-11708				
TSFP27	056464	#7-11811				
TSFP3	052430	#7-10785				
TSFP30	056772	#7-11907				
TSFP31	057302	#7-12003				
TSFP32	057422	#7-12048				
TSFP33	057674	#7-12130				
TSFP4	052524	#7-10820				
TSFP5	052750	#7-10890				
TSFP6	053040	#7-10921				
TSFP7	053214	#7-10976				
TSF10	053620	7-11046	#7-11084			
TSF13	054130	7-11160	#7-11190			
TSF14	054260	7-11201	#7-11232			
TSF15	054446	7-11248	#7-11290			
TSF16	054612	7-11306	#7-11342			
TSF17	054774	7-11358	#7-11399			
TSF2	052404	7-10760	7-10762	7-10764	7-10766	#7-10771
TSF20	055146	7-11416	#7-11452			
TSF21	055336	7-11469	#7-11510			
TSF22	055502	7-11527	#7-11559			
TSF23	055622	7-11576	#7-11601			
TSF24	055776	7-11617	#7-11653			
TSF31	057414	7-12007	#7-12035			
TSF6	053210	7-10928	#7-10969			
TSF7	053400	7-10982	#7-11020			
TSGPR0	= ******	7-1308				
TSGPR1	= ******	7-1334				
TSGPR2	= ******	7-1360				
TSGPR3	= ******	7-1386				
TSGPR4	= ******	7-1412				
TSGPR5	= ******	7-1438				
TSGPR6	= ******	7-1464				
TSLOOP	112072	#27-18551	27-18627	27-18632		
TSMA	043076	7-9234	#7-9237			
TSMB	043116	7-9239	#7-9242			
TSMC	043126	7-9243	#7-9246			
TSMMU0	015036	#7-3820				
TSMMU1	035132	#7-8088				
TSMMU2	035216	#7-8108				
TSMMU3	036416	#7-8364				
TSMMU4	036716	#7-8439				
TSMMU5	037646	#7-8627				
TSMMU6	040000	#7-8660				
TSMMU7	042504	#7-9144				
TSMMU8	043132	#7-9251				
TSMMU9	043334	#7-9294				
TSMM10	044234	#7-9506				
TSMM11	044616	#7-9590				
TSMM12	045054	#7-9641				
TSMM13	045442	#7-9727				
TSMM14	046510	#7-9959				

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
TSMM15	047634	#7-10203
TSMM16	050154	#7-10302
TSMM16A	040106	#7-8683
TSMM16B	040544	#7-8787
TSMM16C	041304	#7-8916
TSMM16D	042004	#7-9030
TSM16A	050250	#7-10319 7-10351
TSM16B	050276	#7-10325 7-10359
TSM16C	050322	#7-10331 7-10367
TSM16D	050366	#7-10339 7-10392
TSM7	043064	7-9177 7-9189 #7-9233
TSM9	043470	7-9307 7-9317 #7-9335 7-9375
TSPSWB	- ******	7-1491
TSTADD	003024	#7-993
TSTEND	114304	32-19014 32-19078 #32-19091
TSTLOC	003162	#7-1025 7-10942 7-10944 7-10988 7-10991 7-11000 7-11004 7-11052 7-11055
		7-11064 7-11068 7-11113 7-11165 7-11168 7-11171 7-11206 7-11209 7-11212
		7-11216 7-11253 7-11256 *7-11256 7-11257 7-11260 7-11264 7-11265 7-11311
		7-11314 7-11317 7-11363 7-11366 *7-11366 7-11367 7-11370 7-11374 7-11420
		7-11423 7-11426 7-11430 7-11473 7-11476 *7-11476 7-11477 7-11480 7-11484
		7-11491 7-11531 7-11534 *7-11534 *7-11535 7-11537 7-11580 *7-11583 7-11586
		7-11621 7-11624 *7-11624 7-11625 7-11627 7-11634 7-11672 7-11675 7-11678
		7-11682 7-11817 7-11821 7-11824 7-11834 7-11837 7-11841 7-11848 7-11851
		7-11855 7-11913 7-11917 7-11920 7-11930 7-11933 7-11937 7-11944 7-11947
		7-11951 7-12054 *7-12058 7-12060 *7-12067 7-12069 *7-12076 7-12078 7-12136
		*7-12140 7-12142 *7-12149 7-12151 *7-12158 7-12160 *7-12292 7-12293 *7-12294
		7-12295 7-12305 7-12309 7-12381 *7-12382 7-12392 *7-12452 7-12455 7-14998
		7-15004 7-15038 7-15044 7-15078 7-15079 *7-15079 *7-15080 7-15081 *7-15082
		7-15086 7-15090 7-15120 *7-15121 7-15126 7-15160 7-15161 *7-15161 *7-15162
		7-15163 *7-15164 7-15168 7-15172 7-15202 *7-15203 7-15204 7-15208 7-15212
		7-15216 7-15242 7-15243 *7-15243 *7-15244 7-15245 7-15246 *7-15246 7-15250
		7-15254 7-15258 *9-16432 *9-16434 9-16438 9-16478 *9-16494 *9-16538 *9-16540
		9-16543 9-16548 10-16605 10-16606 10-16614 *10-16619 10-16623 11-16750 *11-16752
		11-16753 *11-16758 *11-16759 11-16765 *11-16768 *11-16769 11-16774 *12-16841 *12-16845
		12-16849 12-16856 12-16859 12-16865 14-17013 14-17014 14-17017 *14-17022 14-17027
		*14-17032 *15-17123 *15-17126 15-17129 *15-17143 15-17145 *16-17219 16-17221 *17-17297
		17-17300 *18-17384 18-17387 *19-17449 19-17451 *19-17465 19-17467 *20-17581 20-17583
		20-17589 20-17599 *20-17601 20-17603 20-17610 *20-17612 20-17614 20-17627 *20-17629
		20-17631 20-17637 *20-17639 20-17641 22-17800 22-17803 22-17814 *22-17823 58-21193
TSTLUP	110724	#26-18292 26-18374 26-18379
TST1	004724	#7-1195
TST10	103446	11-16744 11-16780 #12-16833
TST11	103710	12-16838 #13-16918
TST12	104164	13-16923 13-16965 #14-17005
TST13	104364	14-17010 #15-17110
TST14	104700	15-17115 15-17162 #16-17207
TST15	105044	16-17212 16-17229 #17-17276
TST16	105272	17-17281 17-17308 #18-17364
TST17	105476	18-17369 18-17394 #19-17430
TST2	101716	#7-16270
TST20	105722	19-17435 #20-17565
TST21	106470	20-17651 #21-17721

M6

SYMBOL	VALUE	REFERENCES						
TST22	106700	21-17726	21-17752	#22-17791				
TST23	107114	22-17796	22-17830	#23-17889				
TST24	107452	23-17894	#24-18011					
TST25	110230	24-18093	#25-18109					
TST26	110372	25-18114	#26-18230					
TST27	111506	#27-18483						
TST3	102062	7-16276	7-16300	#8-16328				
TST30	112630	#28-18668						
TST31	113002	28-18706	#29-18742					
TST32	113250	29-18793	#30-18836					
TST33	113570	#31-18922						
TST34	113722	#32-19004						
TST35	114306	#33-19125						
TST36	114472	33-19130	33-19160	#34-19210				
TST37	115000	34-19214	#35-19307					
TST4	102162	8-16333	#9-16422					
TST40	115146	35-19311	#36-19378					
TST41	115450	36-19382	#37-19459					
TST42	115616	37-19461	37-19465	37-19486	#38-19525			
TST43	116126	38-19536	38-19585	#39-19604				
TST44	116166	39-19612	#40-19645					
TST45	116264	#41-19685						
TST46	116346	41-19690	41-19699	#42-19729				
TST47	116562	42-19734	42-19742	#43-19790				
TST5	102546	9-16427	9-16495	#9-16528				
TST50	116614	43-19792	43-19799	#44-19832				
TST51	117160	#45-19941						
TST52	117426	45-19945	45-19971	#46-20034				
TST53	117716	46-20039	46-20041	46-20067	46-20073	#47-20120		
TST54	120134	47-20124	47-20146	#48-20175				
TST55	120456	48-20209	#49-20256					
TST56	121032	49-20258	49-20260	49-20262	49-20319	#49-20343		
TST57	121300	#50-20415						
TST6	102732	9-16534	9-16555	#10-16597				
TST60	121452	50-20420	#51-20468					
TST61	121652	51-20473	51-20480	#52-20535				
TST62	122620	#53-20716						
TST63	123076	53-20718	53-20720	#54-20798				
TST64	123262	54-20800	54-20802	#55-20851				
TST65	123362	55-20853	55-20855	#56-20889				
TST66	123544	56-20891	#57-20961					
TST7	103172	10-16602	10-16633	#11-16739				
TS10	044534	7-9522	7-9533	7-9544	7-9558	#7-9566		
TS10D1	053624	7-11047	7-11059	#7-11088				
TS10D2	053634	7-11049	7-11056	#7-11092				
TS10D4	053644	7-11067	#7-11096					
TS11	044734	7-9598	7-9603	7-9608	#7-9614			
TS11D1	053740	7-11109	#7-11129					
TS12	045372	7-9659	7-9666	7-9673	7-9680	7-9687	7-9694	#7-9701
TS14	047114	7-10010	7-10032	#7-10043				
TS15	050000	7-10210	7-10218	7-10227	#7-10252			
TS16	051004	7-10430	#7-10437					

SYMBOL	VALUE	REFERENCES								
TS16A	050776	#7-10434	7-10449							
TS1822	046046	#7-9854								
TS26D0	056402	7-11713	7-11719	7-11742	7-11748	#7-11778				
TS26D1	056412	7-11722	7-11728	7-11751	7-11757	#7-11782				
TS26D2	056422	7-11731	7-11737	7-11760	#7-11786					
TS26D3	056432	7-11720	7-11749	#7-11790						
TS26D4	056442	7-11729	7-11758	#7-11794						
TS26D5	056452	7-11738	#7-11798							
TS27D0	056710	7-11818	#7-11874							
TS27D1	056720	#7-11878								
TS27D2	056730	#7-11882								
TS27D3	056740	7-11828	#7-11886							
TS27D4	056750	7-11842	#7-11890							
TS27D5	056760	7-11856	#7-11894							
TS30D0	057220	7-11914	#7-11970							
TS30D1	057230	#7-11974								
TS30D2	057240	#7-11978								
TS30D3	057250	7-11924	#7-11982							
TS30D4	057260	7-11938	#7-11986							
TS30D5	057270	7-11952	#7-11990							
TS32D0	057612	7-12055	#7-12097							
TS32D1	057622	#7-12101								
TS32D3	057642	7-12061	#7-12109							
TS32D4	057652	7-12070	#7-12113							
TS32D5	057662	7-12079	#7-12117							
TS33D0	060066	7-12137	#7-12179							
TS33D1	060076	#7-12183								
TS33D2	060106	#7-12187								
TS33D3	060116	7-12143	#7-12191							
TS33D4	060126	7-12152	#7-12195							
TS33D5	060136	7-12161	#7-12199							
TS6DA	053170	7-10935	7-10948	#7-10960						
TS6DAT	053200	7-10929	7-10931	#7-10964						
TS7	042754	7-9157	7-9164	#7-9205						
TS7DA1	053404	7-10983	7-10995	#7-11024						
TS7DA2	053414	7-10985	7-10992	#7-11028						
TS7DA4	053424	7-11003	#7-11032							
TS7FIN	043130	7-9201	#7-9248							
TS9FIN	044232	7-9331	#7-9503							
TT	= ******	7-7112	7-7141							
TTR	= ******	7-7170								
TYPDS	= 104405	49-20311	49-20317	64-21664	#64-21673					
TYPE	= 104401	7-1162	32-19019	48-20185	48-20186	48-20203	49-20312	49-20318	59-21226	59-21243
		59-21245	59-21248	59-21250	59-21254	59-21273	62-21474	64-21664	64-21664	64-21666
		64-21666	64-21668	64-21669	64-21670	64-21671	64-21671	64-21671	64-21671	64-21671
		64-21671	64-21671	64-21671	64-21671	#64-21673	64-21674			
TYPOC	= 104402	59-21234	59-21270	64-21671	#64-21673					
TYPON	= 104404	#64-21673								
TYPOS	= 104403	#64-21673								
T10FIN	044616	7-9561	#7-9587							
T11FIN	045054	7-9610	#7-9638							
T114	021126	7-5075	7-5085	#7-5093						

SYMBOL	VALUE	REFERENCES			
T116	022572	7-5465	7-5474	#7-5504	7-5515
T12FIN	045440	7-9697	#7-9717		
T122A	023652	7-5747	7-5758	#7-5768	
T122B	024004	7-5745	#7-5807		
T123A	024210	7-5824	#7-5867		
T123B	024220	#7-5871			
T123C	024226	#7-5874			
T123D	024234	7-5828	#7-5877		
T123E	024236	7-5860	#7-5879		
T123F	024242	7-5864	#7-5881		
T124A	024442	7-5891	#7-5939		
T124B	024452	#7-5943			
T124C	024460	#7-5946			
T124D	024466	7-5895	#7-5949		
T124E	024470	7-5932	#7-5951		
T124F	024474	7-5936	#7-5953		
T13FIN	046046	7-9773	#7-9843		
T14	047140	7-10028	#7-10048		
T14FIN	047512	7-9861	7-10037	7-10070	#7-10160
T15	050050	7-10213	7-10221	7-10230	#7-10269
T15A	050126	7-10235	7-10239	7-10243	#7-10288
T15FIN	050154	7-10247	#7-10298		
UDPAR0	= 177660	#5-441			
UDPAR1	= 177662	#5-441			
UDPAR2	= 177664	#5-441			
UDPAR3	= 177666	#5-441			
UDPAR4	= 177670	#5-441			
UDPAR5	= 177672	#5-441			
UDPAR6	= 177674	#5-441			
UDPAR7	= 177676	#5-441			
UDPDR0	= 177620	#5-441	*7-8699	*7-8931	
UDPDR1	= 177622	#5-441			
UDPDR2	= 177624	#5-441			
UDPDR3	= 177626	#5-441			
UDPDR4	= 177630	#5-441			
UDPDR5	= 177632	#5-441			
UDPDR6	= 177634	#5-441			
UDPDR7	= 177636	#5-441			
UFDFLG	004116	*7-1148	#7-1148	7-1150	64-21666
UFDSET	= 000001	#5-443	7-1148	64-21666	64-21666
UIPAR0	= 177640	#5-441			
UIPAR1	= 177642	#5-441			
UIPAR2	= 177644	#5-441			
UIPAR3	= 177646	#5-441			
UIPAR4	= 177650	#5-441			
UIPAR5	= 177652	#5-441			
UIPAR6	= 177654	#5-441			
UIPAR7	= 177656	#5-441			
UIPDR0	= 177600	#5-441	*7-8801	*7-9045	
UIPDR1	= 177602	#5-441			
UIPDR2	= 177604	#5-441			
UIPDR3	= 177606	#5-441			

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
UIPDR4	= 177610	#5-441								
UIPDR5	= 177612	#5-441								
UIPDR6	= 177614	#5-441								
UIPDR7	= 177616	#5-441								
UNXPIR	034430	7-7810	#7-7881							
UQUIET	004120	*7-1148	#7-1148	*7-1150	64-21666	64-21666				
UVAD	= ******	7-13046								
VIREOP	124272	#57-21055	64-21664							
VIR1	045730	7-9741	#7-9803							
VIR2	047342	7-10007	#7-10105							
VIR3	047460	7-9987	7-10009	#7-10144						
VMXOR	004122	7-1148	7-1148	7-1148	#7-1148					
VQBE1	002676	#7-927	53-20725	53-20754	54-20803	56-20898				
VQBE2	002716	#7-937	56-20900							
VQPR1	002700	#7-928	53-20726	53-20755	54-20804	56-20899				
VQPR2	002720	#7-938	56-20901							
WC	002672	#7-925	52-20595	52-20631	57-20983	62-21504	62-21518	62-21523		
WC2	002712	#7-935	57-20984							
WLDTRP	135360	7-13110	7-13161	#63-21614						
XBUF	= 177566	#7-964	40-19662	*41-19697	*42-19755	*44-19881	*45-19978	*46-20080	*47-20128	*47-20133
		*47-20152								
XCBIT	016670	#7-4382								
XCSR	= 177564	#7-965	41-19691	41-19694	41-19698	42-19735	*42-19737	42-19738	*42-19740	*42-19762
		*42-19768	*42-19772	*42-19774	*43-19793	43-19798	*44-19839	44-19840	44-19848	*44-19859
		*44-19873	44-19879	*44-19886	*44-19891	*44-19907	*45-19953	*45-19966	*45-19969	*45-19977
		*45-19983	*45-19997	*46-20045	*46-20048	46-20049	*46-20057	*46-20065	*46-20068	46-20069
		*46-20071	46-20078	*46-20086	*46-20088	*47-20125	47-20126	47-20131	*47-20140	*47-20144
		47-20150	*47-20157	*47-20159						
XDF1	= ******	7-12733								
XME100	017674	#7-4728								
XME101	017716	#7-4741								
XOR	= ******	7-4613								
XRCC	= ******	7-4517								
$APTHD	000232	5-453	#5-453							
$ASTAT	= ******	64-21667	64-21667							
$ATYC	136512	64-21667	#64-21667							
$ATY1	136466	#64-21667								
$ATY3	136474	#64-21667	64-21668							
$ATY4	136504	64-21666	#64-21667							
$AUTOB	001134	#6-454	*7-1162	64-21671	64-21671	64-21671				
$BASE	001254	#6-454								
$BDADR	001122	#6-454	*7-16285	*7-16291	*7-16297	*9-16498	*11-16783	*15-17165	*16-17232	*17-17314
		*18-17399	*23-17914	*23-17915	*23-17939	*23-17940	*24-18057	*24-18058	*26-18270	*26-18271
		*27-18522	*27-18523	*32-19051	58-21192	58-21199	58-21200	58-21201	58-21208	58-21209
		58-21210	*62-21403	*62-21531						
$BDDAT	001126	#6-454	*22-17810	*22-17821	*22-17831	*32-19049	*40-19660	*42-19764	42-19765	*46-20062
		46-20063	*46-20083	46-20084	*47-20136	47-20138	47-20142	58-21198	58-21206	58-21208
		58-21210								
$BELL	001170	#6-454	48-20203	64-21666	64-21666	64-21666				
$CDW1	001260	#6-454								
$CDW2	001262	#6-454								
$CHARC	137264	*64-21668	*64-21668	64-21668	*64-21668	#64-21668				

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
$CKSWR	137742	#64-21671	64-21673	64-21673						
$CMTAG	001100	#6-454	7-1149	7-1149	7-1149	7-1149	7-1149	7-1149	7-1149	
$CM3	= 000000	#6-454	6-454							
$CM4	= 000002	#6-454	6-454	6-454	#6-454	6-454	6-454	#6-454		
$CNTLG	140477	64-21671	#64-21671							
$CNTLU	140472	64-21671	#64-21671							
$CPUOP	001226	#6-454								
$CRLF	001175	#6-454	48-20185	59-21226	59-21245	59-21250	59-21254	64-21666	64-21666	64-21666
		64-21668	64-21668	64-21668	64-21671	64-21671	64-21671			
$DBLK	137732	64-21670	64-21670	#64-21670						
$DDW0	001264	#6-454								
$DDW1	001266	#6-454								
$DDW10	001310	#6-454								
$DDW11	001312	#6-454								
$DDW12	001314	#6-454								
$DDW13	001316	#6-454								
$DDW14	001320	#6-454								
$DDW15	001322	#6-454								
$DDW2	001270	#6-454								
$DDW3	001272	#6-454								
$DDW4	001274	#6-454								
$DDW5	001276	#6-454								
$DDW6	001300	#6-454								
$DDW7	001302	#6-454								
$DDW8	001304	#6-454								
$DDW9	001306	#6-454								
$DEVCT	001210	#6-454								
$DEVM	001256	#6-454								
$DOAGN	135556	64-21664	64-21664	#64-21664						
$DTBL	137722	64-21670	#64-21670							
$ENDAD	135546	5-452	7-1162	#64-21664	64-21666	64-21666				
$ENDCT	135514	7-1149	#64-21664							
$ENDMG	135565	64-21664	#64-21664							
$ENULL	135562	64-21664	#64-21664							
$ENV	001220	#6-454	7-1162	7-1165	41-19687	42-19731	45-19942	47-20121	48-20182	49-20261
		57-21055	64-21666	64-21667	64-21667	64-21668				
$ENVM	001221	#6-454	7-1149	64-21667	64-21668	64-21668				
$EOP	135446	51-20478	52-20545	52-20548	56-20894	57-20971	57-20974	57-21053	#64-21664	
$EOPCT	135506	*7-1149	#64-21664	64-21664						
$ERFLG	001103	#6-454	64-21665	64-21665	64-21665	*64-21665	64-21665	64-21665	*64-21666	64-21666
		64-21666								
$ERMAX	001115	#6-454	*7-1149	64-21665	*64-21665	64-21665	64-21665			
$ERROR	136104	7-1149	7-7433	7-7462	#64-21666					
$ERRPC	001116	#6-454	58-21189	58-21190	58-21191	58-21192	58-21193	58-21194	58-21195	58-21196
		58-21197	58-21198	58-21199	58-21200	58-21201	58-21202	58-21203	58-21204	58-21205
		58-21206	58-21207	58-21208	58-21210	59-21232	*64-21666	*64-21666	64-21666	64-21666
		64-21666								
$ERRTB	001324	#7-454	59-21240							
$ERTTL	001112	#6-454	*64-21666	64-21666	64-21666					
$ESCAP	001166	#6-454	*7-1149	*64-21665	64-21666	64-21666	64-21666	64-21666		
$ETABL	001220	#6-454								
$ETEND	001324	5-453	#6-454							

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
$FATAL	001202	#6-454 *64-21667
$FFLG	136732	*64-21667 *64-21667 64-21667 *64-21667 #64-21667
$FILLC	001156	#6-454 64-21668 64-21668 64-21668
$FILLS	001155	#6-454 64-21668 64-21668
$GDADR	001120	#6-454
$GDDAT	001124	#6-454 *9-16436 *10-16626 *12-16842 *17-17320 *18-17408 *20-17586 *20-17606 *20-17617
		*20-17634 *20-17644 *28-18681 *28-18682 *28-18693 *28-18694 *28-18704 *29-18749 *29-18752
		*29-18757 *29-18768 *29-18776 *29-18782 *34-19254 *34-19268 *35-19319 *35-19324 *36-19391
		*36-19396 *36-19418 *42-19767 *47-20137 *49-20392 58-21191 58-21193 58-21194 58-21197
		58-21202 58-21203 58-21204 58-21205 58-21206
$GET42	135536	64-21664 #64-21664
$GTSWR	140012	#64-21671 64-21673 64-21673
$HD	= 000001	5-436 5-436 5-436
$HIBTS	000232	#5-453
$HIOCT	140626	*64-21672 #64-21672
$ICNT	001104	#6-454 *64-21665 64-21665 *64-21665 64-21665 64-21665
$ILLUP	141052	64-21674 64-21674 #64-21674
$INTAG	001135	#6-454 64-21671 64-21671 64-21671
$ITEMB	001114	#6-454 59-21229 *64-21666 64-21666 64-21666 64-21666
$LF	001176	#6-454 64-21666 64-21666 64-21668 64-21668 64-21671 64-21671 64-21671
$LFLG	136731	*64-21667 #64-21667
$LPADR	001106	#6-454 *7-1149 *64-21665 *64-21665 64-21665 64-21665 64-21665
$LPERR	001110	#6-454 *7-1149 64-21665 *64-21665 64-21665 64-21665 64-21666
$MADR1	001232	#6-454
$MADR2	001236	#6-454
$MADR3	001242	#6-454
$MADR4	001246	#6-454
$MAIL	001200	5-453 5-453 #6-454 7-1149 7-1162 64-21665 64-21666 64-21668
$MAMS1	001230	#6-454
$MAMS2	001234	#6-454
$MAMS3	001240	#6-454
$MAMS4	001244	#6-454
$MBADR	000234	#5-453
$MFLG	136730	*64-21667 64-21667 *64-21667 #64-21667
$MNEW	140515	64-21671 #64-21671
$MSGAD	001214	#6-454 *64-21667 64-21667
$MSGLG	001216	#6-454 *64-21667
$MSGTY	001200	#6-454 64-21667 *64-21667 64-21667 *64-21667
$MSWR	140504	64-21671 #64-21671
$MTYP1	001231	#6-454
$MTYP2	001235	#6-454
$MTYP3	001241	#6-454
$MTYP4	001245	#6-454
$MXCNT	136102	64-21665 64-21665 64-21665 #64-21665
$NULL	001154	#6-454 64-21668 64-21668 64-21668
$NWTST	= 000000	#7-1195 #7-16270 #8-16328 #9-16422 #9-16528 #10-16597 #11-16739 #12-16833 #13-16918
		#14-17005 #15-17110 #16-17207 #17-17276 #18-17364 #19-17430 #20-17565 #21-17721 #22-17791
		#23-17889 #24-18011 #25-18109 #26-18230 #27-18483 #28-18668 #29-18742 #30-18836 #31-18922
		#32-19004 #33-19125 #34-19210 #35-19307 #36-19378 #37-19459 #38-19525 #39-19604 #40-19645
		#41-19685 #42-19729 #43-19790 #44-19832 #45-19941 #46-20034 #47-20120 #48-20175 #49-20256
		#49-20343 #50-20415 #51-20468 #52-20535 #53-20716 #54-20798 #55-20851 #56-20889 #57-20961
$OCNT	137512	*64-21669 *64-21669 #64-21669

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES
$OMODE	137514	*64-21669 *64-21669 64-21669 *64-21669 *64-21669 #64-21669
$OVER	136060	64-21665 64-21665 64-21665 64-21665 #64-21665
$PASS	001206	#6-454 *7-1149 7-5653 29-18784 32-19006 37-19460 41-19689 42-19733 43-19791 44-19833 45-19944 46-20040 47-20123 48-20180 49-20259 *64-21664 *64-21664 64-21664 64-21664 64-21664 64-21665 64-21665 64-21665
$PASTM	000240	#5-453
$POWER	141060	32-19025 32-19069 64-21674 #64-21674
$PWRDN	140712	7-1149 #64-21674 64-21674
$PWRMG	141046	#64-21674
$PWRUP	140764	64-21674 #64-21674
$QUES	001174	#6-454 64-21666 64-21666 64-21668 64-21668 64-21671 64-21671 64-21671 64-21671
$RDCHR	140224	#64-21671 64-21673 64-21673
$RDDEC	= ******	64-21673
$RDLIN	140354	#64-21671 64-21673 64-21673
$RDOCT	140526	#64-21672 64-21673 64-21673
$RDSZ	= 000010	#64-21671 64-21671
$RTNAD	135560	#64-21664
$R2A	= ******	64-21673
$SAVRE	= ******	64-21673
$SAVR6	141056	*64-21674 64-21674 *64-21674 *64-21674 #64-21674
$SCOPE	135602	7-1149 #64-21665
$SETUP	= 000137	#7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1148 #7-1148 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1149 7-1162 7-1162 7-1162 64-21664 64-21664 64-21665 64-21666 64-21666 64-21666 64-21666 64-21671 64-21671
$STUP	= 177777	#7-1148 #7-1148 7-1148 #7-1148 #7-1148 7-1148 #7-1148 #7-1148 7-1148 #7-1148 #7-1148 7-1148 #7-1148 #7-1148 7-1148
$SVLAD	136024	64-21665 #64-21665
$SVPC	= 000232	#5-452 5-452
$SWR	= 167400	#5-434 5-436 5-437 5-437 5-437 5-437 5-437 5-437 5-437 5-437 6-454 6-454 6-454 7-1149 7-1149 7-1149 7-1149 7-1149 7-1195 7-16270 8-16328 9-16422 9-16528 10-16597 11-16739 12-16833 13-16918 14-17005 15-17110 16-17207 17-17276 18-17364 19-17430 20-17565 21-17721 22-17791 23-17889 24-18011 25-18109 26-18230 27-18483 28-18668 29-18742 30-18836 31-18922 32-19004 33-19125 34-19210 35-19307 36-19378 37-19459 38-19525 39-19604 40-19645 41-19685 42-19729 43-19790 44-19832 45-19941 46-20034 47-20120 48-20175 49-20256 49-20343 50-20415 51-20468 52-20535 53-20716 54-20798 55-20851 56-20889 57-20961 64-21664 64-21664 64-21664 64-21664 64-21664 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21666 64-21666 64-21666 64-21666 64-21666 64-21666 64-21666 64-21666 64-21666 64-21666 64-21674
$SWREG	001222	#6-454 7-1149
$SWRMK	= 000300	#5-435 5-437 5-437 5-437 5-437 5-437 5-437 5-437 5-437 5-437 5-437 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665 64-21665
$TESTN	001204	#6-454 *7-9859 *64-21665
$TIMES	001164	#6-454 *7-1149 *64-21664 *64-21665 64-21665 *64-21665 64-21665 64-21665
$TKB	001146	#6-454 64-21668 64-21668 64-21668 64-21668 64-21671 64-21671 64-21671 64-21671 64-21671 64-21671
$TKS	001144	#6-454 64-21668 64-21668 64-21668 64-21668 64-21671 64-21671 64-21671 64-21671

SYMBOL	VALUE	REFERENCES
		64-21671 64-21671 64-21671
$TMP0	001160	#5-446 #5-449 #6-454 #23-17917 23-17918 #23-17942 23-17943 #24-18019 24-18091
		#26-18246 26-18277 #27-18499 27-18528 #30-18846 #30-18851 30-18863 #30-18875 #30-18880
		30-18897 #31-18926 #31-18935 31-18945 #33-19134 33-19142 #51-20491 51-20511 #53-20734
		#53-20736 53-20737 53-20747 #53-20763 #53-20765 53-20766 53-20776 #55-20856 55-20867
		#59-21260 #59-21261 #59-21262 #59-21263 59-21264 #62-21432 62-21445 #62-21461 62-21462
$TMP1	001162	#6-454 #10-16605 10-16619 #31-18944 58-21189 58-21190 58-21191 58-21192 58-21193
		58-21194 58-21195 58-21196 58-21197 58-21198 58-21199 58-21200 58-21201 58-21202
		58-21203 58-21204 58-21205 58-21206 58-21207 58-21208 58-21209 58-21210 #59-21224
		#59-21225
$TN	= 000067	5-436 #5-436 7-1195 7-1195 #7-1195 7-16270 7-16270 #7-16270 7-16276
		7-16300 8-16328 8-16328 #8-16328 8-16333 9-16422 9-16422 #9-16422 9-16427
		9-16495 9-16528 9-16528 #9-16528 9-16534 9-16555 10-16597 10-16597 #10-16597
		10-16602 10-16633 11-16739 11-16739 #11-16739 11-16744 11-16780 12-16833 12-16833
		#12-16833 12-16838 13-16918 13-16918 #13-16918 13-16923 13-16965 14-17005 14-17005
		#14-17005 14-17010 15-17110 15-17110 #15-17110 15-17115 15-17162 16-17207 16-17207
		#16-17207 16-17212 16-17229 17-17276 17-17276 #17-17276 17-17281 17-17308 18-17364
		18-17364 #18-17364 18-17369 18-17394 19-17430 19-17430 #19-17430 19-17435 20-17565
		20-17565 #20-17565 20-17651 21-17721 21-17721 #21-17721 21-17726 21-17752 22-17791
		22-17791 #22-17791 22-17796 22-17830 23-17889 23-17889 #23-17889 23-17894 24-18011
		24-18011 #24-18011 24-18093 25-18109 25-18109 #25-18109 25-18114 26-18230 26-18230
		#26-18230 27-18483 27-18483 #27-18483 28-18668 28-18668 #28-18668 28-18706 29-18742
		29-18742 #29-18742 29-18793 30-18836 30-18836 #30-18836 31-18922 31-18922 #31-18922
		32-19004 32-19004 #32-19004 33-19125 33-19125 #33-19125 33-19130 33-19160 34-19210
		34-19210 #34-19210 34-19214 35-19307 35-19307 #35-19307 35-19311 36-19378 36-19378
		#36-19378 36-19382 37-19459 37-19459 #37-19459 37-19461 37-19465 37-19486 38-19525
		38-19525 #38-19525 38-19536 38-19585 39-19604 39-19604 #39-19604 39-19612 40-19645
		40-19645 #40-19645 41-19685 41-19685 #41-19685 41-19690 41-19699 42-19729 42-19729
		#42-19729 42-19734 42-19742 43-19790 43-19790 #43-19790 43-19792 43-19799 44-19832
		44-19832 #44-19832 45-19941 45-19941 #45-19941 45-19945 45-19971 46-20034 46-20034
		#46-20034 46-20039 46-20041 46-20067 46-20073 47-20120 47-20120 #47-20120 47-20124
		47-20146 48-20175 48-20175 #48-20175 48-20209 49-20256 49-20256 #49-20256 49-20258
		49-20260 49-20262 49-20319 49-20343 49-20343 #49-20343 50-20415 50-20415 #50-20415
		50-20420 51-20468 51-20468 #51-20468 51-20473 51-20480 52-20535 52-20535 #52-20535
		53-20716 53-20716 #53-20716 53-20718 53-20720 54-20798 54-20798 #54-20798 54-20800
		54-20802 55-20851 55-20851 #55-20851 55-20853 55-20855 56-20889 56-20889 #56-20889
		56-20891 57-20961 57-20961 #57-20961
$TPB	001152	#6-454 64-21668 64-21668 64-21668
$TPFLG	001157	#6-454 64-21668 64-21668 64-21668
$TPS	001150	#6-454 64-21668 64-21668 64-21668
$TRAP	140630	7-1149 #64-21673
$TRAP2	140652	#64-21673 64-21673
$TRP	= 000013	#64-21673 64-21673 64-21673 64-21673 64-21673 #64-21673 64-21673 64-21673 64-21673
		64-21673 #64-21673 64-21673 64-21673 64-21673 64-21673 #64-21673 64-21673 64-21673
		64-21673 64-21673 #64-21673 64-21673 64-21673 64-21673 64-21673 #64-21673 64-216?3
		64-21673 64-21673 64-21673 #64-21673 64-21673 64-21673 64-21673 64-21673 #64-21673
		64-21673 64-21673 64-21673 64-21673 #64-21673 64-21673 64-21673 64-21673 64-21673
		#64-21673 64-21673 64-21673 64-21673 64-21673 #64-21673
$TRPAD	140664	64-21673 #64-21673
$TSTM	000236	#5-453
$TSTNM	001102	#6-454 59-21225 #64-21664 64-21665 64-21665 #64-21665 64-21665 64-21665 64-21665
		64-21665 64-21666 64-21666 64-21666

SEQ 0499

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES				
$TTYIN	140462	64-21671	64-21671	64-21671	#64-21671	
$TYPBN	= ******	64-21673				
$TYPDS	137516	#64-21670	64-21673	64-21673		
$TYPE	136734	64-21667	#64-21668	64-21673	64-21673	
$TYPEC	137146	64-21668	64-21668	64-21668	#64-21668	64-21671
$TYPEX	137266	64-21668	64-21668	#64-21668		
$TYPOC	137314	#64-21669	64-21673	64-21673		
$TYPON	137330	64-21669	#64-21669	64-21673		
$TYPOS	137270	#64-21669	64-21673			
$UNIT	001212	#6-454				
$UNITM	000242	#5-453				
$USWR	001224	#6-454	7-1167			
$VECT1	001250	#6-454				
$VECT2	001252	#6-454				
$XOFF	= 000023	64-21668	64-21668			
$XON	= 000021	64-21668	64-21668	64-21671		
$XTSTR	135622	#64-21665				
$$GET4	= 000000	#64-21664	64-21664			
$OFILL	137513	*64-21669	*64-21669	64-21669	#64-21669	
$40CAT	= ******	64-21665	64-21666			
.$ASTA	= ******	64-21667	64-21667			
.$X	= 000232	#5-453	5-453			

MACRO NAME	REFERENCES									
ADOPMS	#7-9846	#7-9847	7-9856							
CBITMS	#7-4384	#7-4385	7-4393							
COMMEN	#5-442									
DATA01	#7-2877	#7-2878								
ENDCOM	#5-442									
ESCAPE	#5-442									
FPP1MS	#7-10479	#7-10480	7-10492							
FPP2MS	#7-10626	#7-10627	7-10637							
GETPRI	#5-442									
GETSWR	#5-442	#7-1162	7-1162							
HALTMS	#7-5603	#7-5604								
MFPTMS	#7-5559	#7-5560								
MMMS13	#7-9721	#7-9722	7-9729							
MMMS14	#7-9953	#7-9954	7-9961							
MULT	#5-442									
NEWTST	#5-442	7-1195	7-16270	8-16328	9-16422	9-16528	10-16597	11-16739	12-16833	13-16918
	14-17005	15-17110	16-17207	17-17276	18-17364	19-17430	20-17565	21-17721	22-17791	23-17889
	24-18011	25-18109	26-18230	27-18483	28-18668	29-18742	30-18836	31-18922	32-19004	33-19125
	34-19210	35-19307	36-19378	37-19459	38-19525	39-19604	40-19645	41-19685	42-19729	43-19790
	44-19832	45-19941	46-20034	47-20120	48-20175	49-20256	49-20343	50-20415	51-20468	52-20535
	53-20716	54-20798	55-20851	56-20889	57-20961					
NXPMS	#7-10164	#7-10165	7-10175							
ODDAMS	#7-7701	#7-7702	7-7710							
PIRMS2	#7-7846	#7-7847	7-7857							
PIRMS3	#7-7886	#7-7887	7-7895							
PIRMS4	#7-7923	#7-7924	7-7933							
PIRMS5	#7-7953	#7-7954	7-7968							
PIR6MS	#7-8030	#7-8031	7-8046							
POP	#5-442	64-21667	64-21667	64-21670	64-21672	64-21674	64-21674			
PREBUF	#7-3485	#7-3486	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498
	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498
	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498	7-3498
	7-3498	7-3498	7-3498	7-3498						
PUSH	#5-442	64-21667	64-21667	64-21667	64-21670	64-21672	64-21674	64-21674		
REPORT	#5-442									
SAVEMT	#7-1148	7-1148								
SETPRI	#5-442									
SETTRA	#64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673	64-21673
	64-21673									
SETUP	#5-442	7-1149								
SKIP	#5-432	#5-442	7-16276	7-16300	8-16333	9-16427	9-16495	9-16534	9-16555	10-16602
	10-16633	11-16744	11-16780	12-16838	13-16923	13-16965	14-17010	15-17115	15-17162	16-17212
	16-17229	17-17281	17-17308	18-17369	18-17394	19-17435	20-17651	21-17726	21-17752	22-17796
	22-17830	23-17894	24-18093	25-18114	28-18706	29-18793	33-19130	33-19160	34-19214	35-19311
	36-19382	37-19461	37-19465	37-19486	38-19536	38-19585	39-19612	41-19690	41-19699	42-19734
	42-19742	43-19792	43-19799	45-19945	45-19971	46-20039	46-20043	46-20067	46-20077	47-20124
	47-20146	48-20209	49-20258	49-20260	49-20262	49-20319	50-20420	51-20473	51-20480	53-20718
	53-20720	54-20800	54-20802	55-20853	55-20855	56-20891				
SLASH	#5-442									
SOP1MS	#7-1518	#7-1519	7-1528							
STARS	#5-442	5-452	5-453	5-453	5-453	6-454	6-454	6-454	7-1195	7-1213
	7-1243	7-1528	7-4106	7-4393	7-4567	7-7710	7-7857	7-7895	7-7933	7-7968

MACRO NAME REFERENCES

7-8046 7-9729 7-9856 7-9961 7-10175 7-10492 7-10637 7-16270 8-16328 9-16422
9-16528 10-16597 11-16739 12-16833 13-16918 14-17005 15-17110 16-17207 17-17276 18-17364
19-17430 20-17565 21-17721 22-17791 23-17889 24-18011 25-18109 26-18230 27-18483 28-18668
29-18742 30-18836 31-18922 32-19004 33-19125 34-19210 35-19307 36-19378 37-19459 38-19525
39-19604 40-19645 41-19685 42-19729 43-19790 44-19832 45-19941 46-20034 47-20120 48-20175
49-20256 49-20343 50-20415 51-20468 52-20535 53-20716 54-20798 55-20851 56-20889 57-20961
64-21664 64-21665 64-21666 64-21667 64-21668 64-21669 64-21670 64-21671 64-21671 64-21671
64-21671 64-21672 64-21673 64-21674 64-21674

SWRSU #5-442 #7-1149 7-1149
SXTMSG #7-4556 #7-4557 7-4567
TCC1MS #7-4097 #7-4098 7-4106
TRMTRP #64-21673
TSO1MS #7-1204 #7-1205 7-1213
TSO2MS #7-1234 #7-1235 7-1243
TYPBIN #5-442
TYPDEC #5-442 64-21664
TYPNAM #5-442 7-1162
TYPNUM #5-442
TYPOCS #5-442
TYPOCT #5-442 64-21671
TYPTXT #5-442
$ABORT #64-21666 64-21666
$$CMRE #5-454
$$CMTM #5-454 6-454 6-454
$$ESCA #5-442
$$NEWT #5-442 7-1195 7-16270 8-16328 9-16422 9-16528 10-16597 11-16739 12-16833 13-16918
14-17005 15-17110 16-17207 17-17276 18-17364 19-17430 20-17565 21-17721 22-17791 23-17889
24-18011 25-18109 26-18230 27-18483 28-18668 29-18742 30-18836 31-18922 32-19004 33-19125
34-19210 35-19307 36-19378 37-19459 38-19525 39-19604 40-19645 41-19685 42-19729 43-19790
44-19832 45-19941 46-20034 47-20120 48-20175 49-20256 49-20343 50-20415 51-20468 52-20535
53-20716 54-20798 55-20851 56-20889 57-20961
$$SET #64-21673 64-21673 64-21673 64-21673 64-21673 64-21673 64-21673 64-21673 64-21673 64-21673
64-21673
$$SETM #7-1149 7-1149
$$SKIP #5-442 7-16276 7-16300 8-16333 9-16427 9-16495 9-16534 9-16555 10-16602 10-16633
11-16744 11-16780 12-16838 13-16923 13-16965 14-17010 15-17115 15-17162 16-17212 16-17229
17-17281 17-17308 18-17369 18-17394 19-17435 20-17651 21-17726 21-17752 22-17796 22-17830
23-17894 24-18093 25-18114 28-18706 29-18793 33-19130 33-19160 34-19214 35-19311 36-19382
37-19461 37-19465 37-19486 38-19536 38-19585 39-19612 41-19690 41-19699 42-19734 42-19742
43-19792 43-19799 45-19945 45-19971 46-20039 46-20041 46-20067 46-20073 47-20124 47-20146
48-20209 49-20258 49-20260 49-20262 49-20319 50-20420 51-20473 51-20480 53-20718 53-20720
54-20800 54-20802 55-20853 55-20855 56-20891

.EQUAT #5-429 5-442
.HEADE #5-429 5-436
.KT11 #5-432 5-441
.SETUP #5-429 7-1148
.SWRHI #5-429 5-437
.SWRLO #5-429 #5-437 5-438
.$ACT1 #5-428 5-452
.$APTB #5-428 #6-454 6-454
.$APTH #5-428 5-453
.$APTY #5-428 64-21667

MACRO NAME	REFERENCES	
.$CATC	#5-429	5-444
.$CMTA	#5-430	5-454
.$EOP	#5-430	64-21664
.$ERRO	#5-430	64-21666
.$ERRT	#5-430	
.$POWE	#5-431	64-21674
.$RDOC	#5-431	64-21672
.$READ	#5-431	64-21671
.$SAVE	#5-431	
.$SCOP	#5-430	64-21665
.$SIZE	#5-432	
.$TRAP	#5-431	64-21673
.$TYPD	#5-431	64-21670
.$TYPE	#5-430	64-21668
.$TYPO	#5-431	64-21669