KDF11-B

11/23B CPU CLSTR
CJKDJB0

AH-T039B-MC
FICHE 1 OF 3

KDF11-B

11/23B CPU CLSTR
CJKDJB0

AH-T039B-MC
FICHE 2 OF 3

JUL 1982

digital

KDF11-B

11/23B CPU CLSTR
CJKDJB0

AH-T039B-MC
FICHE 3 OF 3

JUL 1982

digital

```
.REM_

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

                    PRODUCT CODE:  AC-T038B-MC

                    PRODUCT NAME:   CJKDJB0 11/23B CPU CLSTR DIAG

                    PRODUCT DATE:  APRIL-82

                    MAINTAINER:  DIAGNOSTIC ENGINEERING


THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE  WITHOUT  NOTICE
AND  SHOULD  NOT  BE CONSTRUED AS A COMMITMENT BY DIGITAL 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):  1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL        PDP          UNIBUS        MASSBUS
DEC            DECUS        DECTAPE       DEC/X11
```

```
                    HISTORY
                    -------

REVISION A          FIRST RELEASE OF DIAGNOSTIC

REVISION B          CORRECTED MEDIA PROBLEMS
```

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION
1.1 ABSTRACT
1.2 SYSTEM REQUIREMENTS
1.3 RELATED DOCUMENTS AND STANDARDS
1.4 DIAGNOSTIC HIERARCHY PREREQUISITES
1.5 ASSUMPTIONS

2.0 OPERATING INSTRUCTIONS
2.1 LOADING AND STARTING PROCEDURE
2.2 PROGRAM OPTIONS
2.3 EXECUTION TIMES

3.0 ERROR INFORMATION
3.1 ERROR REPORTING PROCEDURES
3.2 ERROR HALTS

4.0 PERFORMANCE AND PROGESS REPORTS
4.1 PERFORMANCE REPORTS
4.2 PROGRESS REPORTS

5.0 DEVICE INFORMATION TABLES

6.0 PROGRAM DESCRIPTION
6.1 PROGRAM EXECUTION CHARACTERISTICS
6.2 SUBTEST SUMMARIES
6.3 SPECIAL SUBROUTINE DESCRIPTION

7.0 LISTING

1.0 ABSTRACT

THIS PROGRAM IS A GO-NOGO TEST FOR THE 11/23-B CPU BOARD. IT TESTS THE CPU INCLUDING EIS, THE MMU, THE FPP, THE LTC AND BOTH SLU'S. IT DOES NOT CONTAIN THE CAPABILITIES OF SCOPE LOOPING, ERROR RECOVERY OR PRINTING OF ERROR INFORMATION. ERROR HALTS DO INDICATE WHICH DEVICE FAILED TO ALLOW THE TECHNICIAN TO DETERMINE WHICH DIAGNOSTIC TO USE TO FIX THE BOARD OR WHAT FIELD REPLACEABLE UNIT (FRU) MAY FIX THE BOARD. THE PROGRAM WILL RUN UNDER THE ACT AND APT MANUFACTURING SYSTEMS AND IS CHAINABLE UNDER XXDP.

1.1 SYSTEM REQUIREMENTS

A. HARDWARE REQUIREMENTS

- KDF11-B CPU MODULE

- 32K OF MEMORY

- THE SECOND SLU MUST HAVE TURN AROUND CONNECTOR.

B. SOFTWARE ENVIRONMENTS

- APT (MULTI-CPU TESTER)

- ACT

- XXDP (SLIDE)

- STAND-ALONE

1.2 RELATED DOCUMENTS AND STANDARDS

- ASSEMBLED WITH SYSMAC; SEE FIRST PAGE OF LISTING FOR REVISION NUMBER.

- MXXXX MODULE SPECIFICATION

- DIAGNOSTIC ENGINEERING FUNCTIONAL SPECIFICATION FOR SPECIAL MANUFACTURING TEST BGI-79-003-00-U.

- DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS 175-003-009-02.

1.3 PREREQUISITE DIAGNOSTICS

NONE

1.4 ASSUMPTIONS

THIS PROGRAM ASSUMES THE MACHINE IS UP SUFFICIENTLY TO ALLOW PROPER OPERATION OF THE MICRO-ODT OF THE DCF11-AA CHIP SET.

THE SYSTEM MUST HAVE PARITY MEMORY LOCATED IN THE FIRST 32K BLOCK.

THE SOFTWARE ASSUMES THAT THERE IS NO MEMORY OR DEVICES LOCATED AT OR BEYOND ADDRESS BIT 17 (64 KW). IF MEMORY IS THERE THE PROGRAM WILL FAIL WHEN IN THE EXTENDED ADDRESS TESTS. IF BIT 7 IS SET IN THE SWITCH REGISTER (176) THIS FAILURE CAN BE PREVENTED SINCE THAT PARTICULAR TEST WILL BE BYPASSED.

SEE PARAGRAPH 2.2.

```
2.0 OPERATING INSTRUCTIONS



2.1 LOADING AND STARTING PROCEDURES

    TO LOAD AND START THIS PROGRAM USE THE  STANDARD  PROCEDURES  FOR
    THE DIAGNOSTIC SOFTWARE ENVIRONMENT THAT IS BEING USED.



2.2 PROGRAM OPTIONS

    THIS PROGRAM USES THE SOFTWARE SWITCH  LOCATION 176 IF PROGRAM IS
    NOT BEING  RUN UNDER  APT MODE (BIT 0 SET OF  LOCATION $ENV).  IF
    PROGRAM IS  BEING RUN IN APT MODE THE LOCATION  $SWREG IN THE APT
    ETABLE IS USED TO STORE OPERATING SWITCHES.

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


WARNING****THIS PROGRAM IS SET TO DO MINIMUM TESTING UNLESS CORRECTIVE
               ACTION IS TAKEN VIA THE SOFTWARE SWITCH REGISTER (176).
               BITS 1, 6,7-10 HAVE BEEN SET UP SUCH THAT THE PROGRAM
               WILL BYPASS CERTAIN TESTS UNLESS THE SWITCH REGISTER
               BIT IS SET. THIS CONDITION ALSO APPLIES WHEN UNDER CONTROL
               OF APT. THE APT SWITCH REGISTER, LOCATION 1022, MUST BE
               CORRECTLY SET AT APT LOAD TIME.


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

    BIT #     DEFINITION

    15-11      NOT USED


     10       1 - TEST E102 SWITCHES
              0 - INHIBIT TESTING E102 SWITCHES

     9        1 - TEST PARITY ERROR DETECTION
              0 - INHIBIT TESTING PARITY ERROR DETECTION

     8        1 - USE THE Q22BE
              0 - USE THE QBE IN PLACE OF THE Q22BE

     7        1 - TEST THE UPPER 5 ADDRESS BITS FOR TIME OUT
              0 - INHIBIT TESTING THE UPPER 5 ADRS BITS

     6        1 - TEST USING A Q BUS EXERCISER (QBE OR Q22BE)
              0 - INHIBIT TESTS THAT USE A Q BUS EXERCISER

     5        0 - PROGRAM RESERVED -- PROGRAM WILL CLEAR IF CIS CHIP SET NOT ON BOARD
              1 - PROGRAM RESERVED -- PROGRAM WILL SET IF CIS CHIP SET IS ON BOARD
```

```
4        0 - TEST SLU2 OF 11/23-B
         1 - INHIBIT TESTING OF SLU2

3        0 - TEST LTC OF 11/23-B
         1 - INHIBIT TESTING OF LTC

2        0 - TEST SLU1 OF 11/23-B
         1 - INHIBIT TESTING OF SLU1

1        1 - TEST FPP INSTRUCTION SET
         0 - INHIBIT TESTING OF FPP

0        0 - TEST MEMORY MANAGEMENT UNIT
         1 - INHIBIT TESTING OF MMEMORY MANAGEMENT UNIT
```

2.3 EXECUTION TIMES

```
FIRST PASS RUNTIME (WORST CASE)............45 SEC
LONGEST TEST TIME..........................30 SEC
ADDITIONAL RUNTIME (EXTRA UNITS)...........NONE
LONGEST PASS TIME..........................45 SEC
```

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURES

THE PROGRAM DOES NOT TYPE OUT ANY ERROR REPORTS OF ITS OWN BUT TAKES ADVANTAGE OF THE HARDWARE FEATURE THAT TYPES THE PC WHEN A HALT OCCURS. WHEN AN ERROR IS DETECTED THE PROGRAM JUMPS TO ONE OF SEVEN HALT ROUTINES. THE ROUTINES SIMPLY MOVE A FATAL ERROR NUMBER INTO LOCATION $FATAL, SET THE FATAL ERROR FLAG IN LOCATION $MSGTY AND EITHER HALT OR IF ON APT DO A BRANCH DOT. THE OPERATOR HAS THREE WAYS TO DETERMINE THE FAILING DEVICE; 1) BY EXAMINING LOCATION $FATAL, 2) BY DETERMINING THE HALT ADDRESS AND LOOKING UP THE ADDRESS IN THE LISTING AND 3) BY EXAMINING LOCATION $TESTN WHICH WILL CONTAIN THE TEST NUMBER BEING EXECUTED.

3.2 ERROR HALTS

FOR DISCUSSION SEE SECTION 3.1. THE LABELS FOR THE HALTS AND THE DEVICE THEY INDICATE HAVING FAILED ARE:

```
CPUHLT:  CPU
MMUHLT:  MMU
FPPHLT:  FPP
LTCHLT:  LTC
SL1HLT:  SLU1
```

```
          SL2HLT:  SLU2
          EXADHT:  EXT ADRS TEST
          COMHLT:  SYSTEM INTERACTION
          Q22HLT:  Q22BE INTERRUPT TEST
          BDVHLT:  BDV TEST

      UPON RECEIVING THE ERROR HALT ADDRESS (PC) FROM THE MICRO-ODT THE
      OPERATOR CAN LOOK UP THESE TAGS IN THE SYMBOL TABLE AT THE END OF
      THE LISTING TO DETERMINE WHICH HALT WAS EXECUTED.  NOTE:  THE  PC
      SUPPLIED BY THE MICRO-ODT WILL BE THE HALT ADDRESS PLUS 2.

      THE OPERATOR CAN DETERMINE WHICH TEST THE ERROR  OCCURRED  IN  BY
      EXAMINING  LOCATION "$TESTN".  THE TEST NUMBERS EQUATE TO FAILING
      DEVICES AS FOLLOWS:

          DEVICE      CONTENTS OF $TESTN

           CPU             000001
           MMU             000002
           FPP             000003
           SLU1            000004
           LTC             000005
           SLU2            000006
           EXTND ADRS      000007
           TESTS
           SYSTEM
           INTERACTION     000010
           Q22BE           000011
           BDV TESTS       000012


4.0   PERFORMANCE AND PROGRESS REPORTS

      THE ONLY REPORT TYPED BY THIS PROGRAM IS  THE  END  PASS  MESSAGE
      WHICH IS:

                CJKDJB END OF PASS #XXX

      WHERE XXX IS THE DECIMAL NUMBER OF PASSES COMPLETED.

5.0   DEVICE INFORMATION TABLES

SLU1 RCSR

       15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
      ---------------------------------------------------------------
      I                                  I   I   I                   I
      ---------------------------------------------------------------
                                           I   I
                                           I   I
 RECEIVER DONE ----------------------------    I
 RECEIVER INTERRUPT ENABLE --------------------
```

```
SLU1 RBUF

     15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
    ----------------------------------------------------------------
   I   I   I   I   I               I   I   I   I   I   I   I   I   I
    ----------------------------------------------------------------
     I   I   I   I                                     I
     I   I   I   I                                     I
ERROR-   I   I   I                                     I
OVERRUN ---  I   I                                     I
FRAME ERROR ---  I                                     I
RECEIVE PARITY ----                                    I
ERROR                                                  I
RECEIVED DATA BITS (8) --------------------------------


SLU1 XCSR

     15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
    ----------------------------------------------------------------
 I                              I   I   I                         I
    ----------------------------------------------------------------
                                 I   I
TRANSMITTER READY ----------------   I                 I
TRANSMITTER INTERRUPT ENABLE ---------                 I

SLU1 XBUF

     15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
    ----------------------------------------------------------------
   I                               I   I   I   I   I   I   I   I   I
    ----------------------------------------------------------------
                                           I
                                           I
TRANSMITTER DATA BITS (8) --------------------


LTC CSR

     15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
    ----------------------------------------------------------------
 I                                      I   I
    ----------------------------------------------------------------
                                          I
                                          I
LINE CLOCK
INTERRUPT ENABLE -------------------------


SLU2 RCSR
```

```
      15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
     ----------------------------------------------------------------
    I                           I   I   I                           I
     ----------------------------------------------------------------
                                  I   I
                                  I   I
RECEIVER DONE ---------------------   I
INTERRUPT ENABLE ----------------------


SLU2 RBUF

      15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
     ----------------------------------------------------------------
    I   I   I   I   I               I   I   I   I   I   I   I   I   I
     ----------------------------------------------------------------
      I   I   I   I                                 I
      I   I   I   I                                 I
ERROR -   I   I   I                                 I
OVERRUN ---   I   I                                 I
FRAME ERROR ---   I                                 I
RECEIVER PARITY ---                                 I
ERROR                                               I
RECEIVER DATA BITS (8) ------------------------------

SLU2 XCSR

      15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
     ----------------------------------------------------------------
                                I   I   I   I                   I   I
     ----------------------------------------------------------------
                                      I   I                       I
                                      I   I                       I
TRANSMITTER DONE ----------------------   I                       I
INTERRUPT ENABLE --------------------------                       I
BREAK ------------------------------------------------------------


SLU2 XBUF
      15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
     ----------------------------------------------------------------
    I                               I   I   I   I   I   I   I   I   I
     ----------------------------------------------------------------
                                                  I
                                                  I
TRANSMITTER DATA BITS (8) -------------------------


6.0  PROGRAM DESCRIPTION
```

6.1 PROGRAM EXECUTION CHARACTERISTICS

THIS PROGRAM RUNS THE SAME UNDER ALL DIAGNOSTIC MONITORS. WHEN THE TEST IS STARTED AT ADDRESS 200 OCTAL THE TESTING IS DONE AND ON COMPLETION THE TITLE IS TYPED AS PART OF THE END OF PASS MESSAGE.

6.2 SUB-TEST SUMMARIES

6.2.1 CENTRAL PROCESSING UNIT SUBTEST -

THESE TESTS CHECK THE BASIC INSTRUCTION SET AND ADDRESSING MODES, THE EXTENDED ELEVEN INSTRUCTION SET (EIS) AND TRAPS TESTING. IT IS EQUIVALENT TO CJKDB.

6.2.2 MEMORY MANAGEMENT UNIT SUBTEST -

THESE TESTS ARE THE SAME AS IN CJKDA, THE KEF11-AA TEST. THE PROGRAM BEGINS BY TESTING SOME OF THE INTERNAL CPU DATA AND ADDRESS PATHS AND ADDRESS DETECTION LOGIC. NEXT THE MEMORY MANAGEMENT REGISTERS ARE CHECKED FOR DATA RELIABILITY, THEN RELOCATION CAPABILITIES (FORMATION OF A PHYSICAL ADDRESS FROM A VIRTUAL ADDRESS AND ASSOCIATED PAGE DESCRIBTOR (PDR) INFORMATION). FINALLY THE ABORT AND STATUS SEGMENTS OF THE LOGIC ARE CHECKED.

6.2.3 EXTENDED ADDRESS BIT TESTING AND PARITY ERROR LOGIC TEST

6.2.4 Q22BE BR LEVEL TESTING

6.2.5 BDV TESTS

6.2.6 FLOATING POINT PROCESSOR SUBTEST -

THE FLOATING POINT PROCESSOR SUBTEST CHECKS FLOATING POINT REGISTERS FIRST USING A LIMITED NUMBER OF FLOATING POINT INSTRUCTIONS. IT THEN VERIFIES THE REST OF THE FLOATING POINT INSTRUCTION SET USING A NUMBER OF DATA PATTERNS FOR EACH INSTRUCTION.

6.2.7 SERIAL LINE UNIT (SLU1) SUBTEST -

THESE TESTS CHECK THE SLU'S REGISTERS FOR ADDRESSING AND DATA HANDLING.

6.2.8 LINE TIME CLOCK (LTC) SUBTEST -

FIRST THE REGISTER IS CHECKED FOR ADDRESSING AND BIT SETTING CAPABILITIES THEN THE INTERRUPT LOGIC IS CHECKED. THERE IS ALSO A

6.2.9 SERIAL LINE UNIT 2 SUBTEST -

THE TESTING DONE HERE IS SIMILIAR AS FOR THE SLU1. AN EXTERNAL JUMPER, TURN AROUND CONNECTOR, MUST BE PRESENT.

6.2.10 BLAST SUBTEST

THIS TEST CHECKS THE ABILITY OF THE 11/23-B TO HANDLE SYSTEM INTERACTION. THE CPU HAS TO HANDLE DEVICES AT DIFFERENT PRIORITY LEVELS AND ARBITRATE BETWEEN THEM AND ITS OWN PRIORITY. THE TEST SETS UP ALL DEVICES TO INTERRUPT THEN ENABLES THEM ALL AT ONCE. THE SLU'S TRANSFER DATA UNTIL THEY TRANSFER 400(8) BYTES OR UNTIL ONE SECOND (60 TICKS) OF THE LINE CLOCK HAS BEEN RECEIVED. THE PROGRAM THEN VERIFIES THE NUMBER OF TRANSMTTER INTERRUPTS IS EQUAL TO THE NUMBER OF RECEIVER INTERRUPTS. FINALLY THE DATA TRANSFERRED BY EACH DEVICE IS CHECKED.

6.3 SPECIAL SUBROUTINE DESCRIPTIONS

THE ONLY SPECIAL SUBROUTINES ARE THE ERROR ROUTINES EACH SUBTEST HAS IS OWN. THE ROUTINES SIMPLY SET THE FATAL ERROR FLAG IN THE APT MAILBOX AND EITHER ''BRANCH SELF'' OR ''HALT''. THIS CHOICE IS DETERMINED IN THE INITIALIZE PORTION OF THE PROGRAM AND IS A ''BRANCH SELF'' IF RUNNING UNDER APT OR A ''HALT'' IF RUNNING UNDER ANY OTHER MONITOR.

-

```
                .TITLE  CJKDJBO  11/23-B CPU CLUSTER DIAG.
                .ENABLE ABS
                .NLIST  CND,MC,MD
                .LIST   ME
000240          SCOPE=NOP
000007          R7=%7
000006          R6=%6
177776          PS=177776
177564          TPS=177564
177566          TPB=177566
140000          USRM=140000
030000          PUSRM=30000
000006          SP=%6
000006          R6=%6
000003          TAB=%3
000001          LAST=%1
000005          FIRST=%5
000002          R2=%2
000000          HLT=HALT
000003          TRT=3
000004          ITRAP5=4
000004          RTRAP5=4                ;RESERVED INST AND ILLEGAL ADDRESSES
000014          RTRAP4=14               ;FOR TRACE TRAP
000030          RTRAP3=30               ;FOR EMULATOR TRAP
000020          RTRAP2=20               ;FOR IOT TRAP
000034          RTRAP1=34               ;FOR TRAP INST
177564          TTCSR=177564
177560          TRCSR=177560
177562          TKB=177562
177566          TPB=177566
```

```
        000240                BELL=240
        000240                NOP=240
        177776                STATUS=177776
        000077                TRAPA=77
        000010                RTRAP=10
        004700                ILLA=004700
        000100                ILLB=100

        177520                PCR=177520
        177524                LSREG=177524
        177522                RWREG=177522

        177776                CC=177776
        001000                KERSTK=STBOT
        000600                USESTK=STBOT-200
        104377                EMTA=104377
        104777                TRAPC=104777
        000001                APTENV=1
                      .SBTTL  ACT11 HOOKS

                      ;*****************************************************************************
                      ;HOOKS REQUIRED BY ACT11
        000400                $SVPC=.                         ;SAVE PC
        000046                .=46
000046  133212                $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                .=52
000052  000000                .WORD   0                       ;;2)SET LOC.52 TO ZERO
        000400                .=$SVPC                         ;; RESTORE PC
        001000                .=1000
                      .SBTTL  APT MAILBOX-ETABLE

                      ;*****************************************************************************
                      .EVEN
001000                $MAIL:                          ;;APT MAILBOX
001000  000000        $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001002  000000        $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001004  000000        $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001006  000000        $PASS:  .WORD   APASS           ;;PASS COUNT
001010  000000        $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001012  000000        $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001014  000000        $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001016  000000        $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001020                $ETABLE:                        ;;APT ENVIRONMENT TABLE
001020     000        $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001021     000        $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001022  000000        $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001024  000000        $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001026  000000        $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                      ;*                               BITS 15-11=CPU TYPE
                      ;*                                       11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                      ;*                                       11/70=06,PDQ=07,Q=10
                      ;*                               BIT 10=REAL TIME CLOCK
                      ;*                               BIT  9=FLOATING POINT PROCESSOR
                      ;*                               BIT  8=MEMORY MANAGEMENT
001030                $ETEND:
                      .MEXIT
```

```
                                .SBTTL  APT PARAMETER BLOCK

                                ;*********************************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;*********************************************************************************
                001030                     .$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  001030                     $APTHDR   ;;POINT TO APT HEADER BLOCK
                001030                     .=.$X     ;;RESET LOCATION COUNTER
                                ;*********************************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

        001030                  $APTHD:
        001030  000000          $HIBTS:  .WORD    0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
        001032  001000          $MBADR:  .WORD    $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
        001034  000010          $TSTM:   .WORD    10         ;;RUN TIM OF LONGEST TEST
        001036  000025          $PASTM:  .WORD    25         ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
        001040  000000          $UNITM:  .WORD    0          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        001042  000014                   .WORD    $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                ;*********************************************************************************
                                ;SOME POINTERS TO CPU TRAP HANDLERS
                                ;*********************************************************************************
                000004                     .=4
        000004  021336                     T04
        000006  000000                     0
        000010  021340                     T010
        000012  000000                     0
        000014  021342                     T014
        000016  000000                     0
        000020  021344                     T020
        000022  000000                     0
                000030                     .=30
        000030  021346                     T030
        000032  000000                     0
        000034  021350                     T034
        000036  000000                     0
                000114                     .=114
        000114  021352                     T0114
        000116  000000                     0
                000244                     .=244
        000244  021354                     T0244
        000246  000000                     0
        000250  021356                     T0250
        000252  000000                     0

                000172          .=172
        000172  000000          MTFLAG:            0        ;MULTI-TESTER ACTIVE BIT
        000174  000000          DISPREG:           0        ;SOFTWARE DISPLAY REGISTER
        000176  000000          SWREG:             0        ;SOFTWARE SWITCH REGISTER

                                ;*********************************************************************************
                                ;DATA TABLE FOR USE IN ADDRESSING MODE TESTS
                                ;*********************************************************************************
```

```
        000370                              .=370
000370  000000  000000  000000              0,0,0,0,0,0
000376  000000  000000  000000
000404  000001  000001  177777              1,1,-1
                                    ;*******************************************************************************
                                    ;SET UP STARTING ADDRESS
        001000                      .=1000
001000  000000                      STBOT:  .WORD 0                             ;STACK POINTER
                                    ;* ****************************************************************


        000510                              .=510                               ;Q22BE DEVICE VECTOR
                                                                                ;AREA

        000200                              .=200
000200  000167  000774                      JMP     START
000204  012706  001000                      MOV     #STBOT,R6                   ;SET STACK POINTER
000210  012702  001004                      MOV     #$TESTN,R2                  ;SET MAILBOX POINTER
000214  000137                              JMP     @(PC)+                      ;JUMP TO SUBTEST
000216  000000                              0                                   ;ADDR. OF SUBTEST GOES HERE

        001200                      .=1200
                                    .SBTTL **STARTING OF CPU TEST **
001200  032737  000001  001020      START:  BIT     #1,@#$ENV                   ;UNDER APT
001206  001004                              BNE     CONTIN
001210  012700  133431                      MOV     #STRMSG,R0                  ;START MESSAGE
001214  004767  132110                      JSR     PC,TYPE
001220  012737  000000  001006      CONTIN: MOV     #0,@#$PASS          ;CLEAR PASS COUNT
001226  012737  133264  000024      RESTRT: MOV     #PWRDN,@#24                 ;SET UP FOR POWER FAIL
001234  012706  001000                      MOV     #STBOT,R6           ;SET UP STACK
001240  012737  001270  000004              MOV     #1$,@#4             ;SET UP FOR TIMEOUT IF NO MULTI TESTER
001246  012737  000340  000006              MOV     #340,@#6
001254  012737  000002  164000              MOV     #2,@#164000         ;SET BIT1 FOR MULTI TESTER
001262  012737  000001  000172              MOV     #1,@#MTFLAG         ;SET FLAG TO INDICATE MULTI-TESTER
001270  012737  021336  000004      1$:     MOV     #T04,@#4                    ;SET TRAP CATCHER
001276  012737  000000  000006              MOV     #0,@#6              ;SET HALT BACK IN LOCATION 6
001304  012706  001000                      MOV     #STBOT,R6           ;INITIALIZE STACK POINTER
001310  012737  000001  001004              MOV     #1,@#$TESTN         ;SET TEST NUMBER TO 1
001316  012737  000000  001002              MOV     #0,@#$FATAL         ;CLEAR ERROR INDICATOR
001324  012737  000000  001000              MOV     #0,@#$MSGTY         ;CLEAR MESSAGE TYPE(FOR APT)

001332  005067  176166                      CLR     LSREG               ;THIS WILL TURN ON THE 4 LEDS

001336  105737  001020                      TSTB    @#$ENV              ;RUNNING ON APT
001342  001036                              BNE     TS1                 ;IF YES DO BRANCH SELF ON ERROR
001344  012737  000000  002164              MOV     #0,@#CPUHLT         ;IF NOT THEN PUT A HALT IN ON ERROR
001352  012737  000000  050470              MOV     #0,@#MMUHLT
001360  012737  000000  051404              MOV     #0,@#EXADHT
001366  012737  000000  053214              MOV     #0,@#Q22HLT
001374  012737  000000  124532              MOV     #0,@#FPHLT
001402  012737  000000  126674              MOV     #0,@#LTCHLT
001410  012737  000000  125754              MOV     #0,@#SL1HLT
001416  012737  000000  131540              MOV     #0,@#SL2HLT
001424  012737  000000  132504              MOV     #0,@#COMHLT
001432  012737  000000  054472              MOV     #0,@#BDVHLT
```

```
                                        ;*************************************************************************************
                                        ;
                                        ;       THIS TEST EXECUTES EVERY POSSIBLE BRANCH WITH EVERY POSSIBLE
                                        ;CONDITION CODE COMBINATION.
                                        ;       THE ROUTINE USES TWO TABLES.  THE BRANCH TABLE HOLDS ALL THE
                                        ;POSSIBLE BRANCH INSTRUCTIONS, THE OTHER TABLE (YNTAB) HOLDS BIT MAPS FOR
                                        ;EACH BRANCH.  A ONE IN THE BIT MAP INDICATES THAT THE CORRESPONDING
                                        ;BRANCH INSTRUCTION SHOULD BRANCH FOR THE CONDITION CODE SETTING WHICH
                                        ;CORRESPONDS TO THE BIT POSITION WITHIN THE MAP.  FOR EXAMPLE IF THE LEFT
                                        ;MOST BIT IS A ONE THEN THE CORRESPONDING BRANCH INSTRUCTION SHOULD BRANCH
                                        ;WHEN THE CONDITION CODES ARE 0.
                                        ;       THE ROUTINE CONSISTS OF NESTED LOOPS; THE OUTER LOOP SETS UP
                                        ;ALL THE POSSIBLE BRANCH INSTRUCTIONS.  THE INNER LOOP SETS UP EVERY POSSIBLE
                                        ;CONDITION CODE FOR EACH BRANCH.
                                        ;       THE BIT MAP IS USED TO SET THE ADDRESS LOCATION IN TWO
                                        ;JUMP MODE 3 INSTRUCTIONS.  THE ADDRESSES ARE CHANGED TO ALLOW THE
                                        ;PROGRAM TO CONTINUE OR JUMP TO AN ERROR ROUTINE DEPENDING UPON
                                        ;WHETHER IT HANDLED THE BRANCH INSTRUCTION CORRECTLY.
                                        ;       AT ANY ERROR HALT, LOCATION, BRH, HOLDS THE BRANCH INSTRUCTION
                                        ;UNDER TEST AND LOCATION, CC, HOLDS THE VALUE OF THE CONDITION CODES
                                        ;AT THE TIME THE BRANCH WAS EXECUTED.
                                        ;
                                        ;*************************************************************************************
                                        ;TEST 1 TEST THE BRANCH ROM
                                        ;*************************************************************************************
001440                                  TS1:
001440  012700  021242                  SETUP:  MOV     #BRTAB,R0       ;INITIALIZE BRANCH TABLE POINTER
001444  012704  021300                          MOV     #YNTAB,R4       ;INITIALIZE YES/NO BRANCH MAP POINTER
001450  012767  000017  000130                  MOV     #15.,BRCT       ;INITIALIZE BRANCH TABLE COUNT
001456  012067  000110                  SETBR:  MOV     (R0)+,BRH       ;GET NEXT BRANCH INST.
001462  012401                                  MOV     (R4)+,R1        ;GET NEXT BRANCH MAP
001464  012767  177777  000074                  MOV     #-1,CC1         ;INITIALIZE CONDITION CODE VALUE
001472  012703  000020                          MOV     #16.,R3         ;INITIALIZE CONDITION CODE COUNT
001476  005267  000064                  SETCC:  INC     CC1             ;SET FOR NEXT CC VALUE
001502  032701  100000                          BIT     #100000,R1      ;SEE IF SHOULD BR W/ THESE CC'S
001506  013705  177776                          MOV     @#177776,R5     ;SIMULATE A JNE
001512  042705  177773                          BIC     #177773,R5      ;       (JUMP NOT EQUAL)
001516  000165  001522                          JMP     .+4(R5)         ;       TO SET2BR
001522  000167  000020                          JMP     SET2BR
001526  012767  001610  000042                  MOV     #CONT,NBR       ;SET TO CONTINUE IF NO BRANCH
001534  012767  001604  000040                  MOV     #ER,YBR         ;SET TO REPORT ERROR IF BRANCH
001542  000167  000014                          JMP     AROUN           ;GO AROUND OPPOSITE CONDITION
001546  012767  001604  000022          SET2BR: MOV     #ER,NBR         ;SET TO REPORT ERROR IF NO BRANCH
001554  012767  001610  000020                  MOV     #CONT,YBR       ;SET TO CONTINUE IF BRANCH
001562  006101                          AROUN:  ROL     R1              ;UPDATE BIT MAP

001564  012737                                  MOV     (PC)+,@(PC)+    ;SET CONDITION CODE
001566  000000                          CC1:    0                       ;NEW CC VALUE GOES HERE
001570  177776                                  177776
001572  000000                          BRH:    0                       ;BRANCH INST. GOES HERE
001574  000137                                  JMP     @(PC)+          ;THIS JUMP IF NO BRANCH
001576  000000                          NBR:    0                       ;WHERE TO GO IF NO BRANCH OCCURS
001600  000137                                  JMP     @(PC)+          ;THIS JUMP IF BRANCH OCCURS
001602  000000                          YBR:    0                       ;WHERE TO GO IF BRANCH OCCURS
001604                                  ER:
001604  000551                                  BR      ERROR1          ;
```

```
001606  000000                      BRCT:   0
001610  005303                      CONT:   DEC     R3              ;CC'S DONE?
001612  013705  177776                      MOV     @#177776,R5     ;SIMULATE A JNE
001616  042705  177773                      BIC     #177773,R5      ;       (JUMP NOT EQUAL)
001622  000165  001626                      JMP     .+4(R5)         ;       TO SETCC
001626  000167  177644                      JMP     SETCC
001632  005367  177750                      DEC     BRCT            ;BR'S DONE?
001636  013705  177776                      MOV     @#177776,R5     ;SIMULATE A JNE
001642  042705  177773                      BIC     #177773,R5      ;       (JUMP NOT EQUAL)
001646  000165  001652                      JMP     .+4(R5)         ;       TO SETBR
001652  000167  177600                      JMP     SETBR
001656  012700  000357                      MOV     #357,R0         ;IF THIS TEST IS DONE SET UP R0 FOR THE NEXT
                                                                    ;SEVEN TESTS. THIS IS SAVING 4 LOCATIONS PER
                                                                    ;TEST WHICH I NEED BECAUSE BRANCHES WERE OUT
                                                                    ;OF BOUNDS.

                                    ;*******************************************************************************
                                    ;TEST 2 TEST TRAP OF RESERVED INSTRUCTION
                                    ;*******************************************************************************
001662                              TS2:
001662  012706  001000                      MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
001666  012767  001710  176114              MOV     #1$,RTRAP               ;SET UP NEW PC IN VECTOR
001674  010067  176112                      MOV     R0,RTRAP+2              ;SET UP NEW PSW IN VECTOR
001700  005067  176072                      CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
001704  000077                              TRAPA                   ;DO TRAP
001706                              8$:
001706  000510                              BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
001710  020067  176062              1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
001714  001105                              BNE     ERROR1          ;NEW STATUS WRONG
001716  020627  000774              2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
001722  001102                              BNE     ERROR1          ;STACK DID NOT DECREMENT CORRECTLY
001724  022767  001706  177042      3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
001732  001076                              BNE     ERROR1          ;PROPER PC WAS NOT SAVED
001734  005767  177036              4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
001740  001073                              BNE     ERROR1          ;WRONG PSW SAVED
001742  062706  000004              5$:     ADD     #4,SP           ;RESET STACK POINTER
001746  012767  001770  176034              MOV     #6$,RTRAP               ;SET UP NEW PC IN VECTOR
001754  005067  176032                      CLR     RTRAP+2         ;SET UP NEW PSW IN VECTOR
001760  010037  177776                      MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
001764  000077                              TRAPA                   ;DO TRAP
001766  000460                              BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
001770  005767  176002              6$:     TST     STATUS          ;IS NEW PSW CORRECT
001774  001055                              BNE     ERROR1          ;NEW PSW WRONG
001776  020067  176774              7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
002002  001052                              BNE     ERROR1          ;OLD STATUS WRONG
                                    ;*******************************************************************************
                                    ;TEST 3 TEST TRAP OF TRAP INSTRUCTION
                                    ;*******************************************************************************
002004                              TS3:
002004  012706  001000                      MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002010  012767  002032  176016              MOV     #1$,RTRAP1              ;SET UP NEW PC IN VECTOR
002016  010067  176014                      MOV     R0,RTRAP1+2             ;SET UP NEW PSW IN VECTOR
002022  005067  175750                      CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
002026  104400                              TRAP                    ;DO TRAP
002030                              8$:
```

```
002030 000437                            BR     ERROR1            ;INSTRUCTION FAILED TO TRAP
002032 020067 175740            1$:      CMP    R0,STATUS         ;IS NEW STATUS CORRECT
002036 001034                            BNE    ERROR1            ;NEW STATUS WRONG
002040 020627 000774            2$:      CMP    SP,#STBOT-4       ;DID STACK DECREMENT CORRECTLY
002044 001031                            BNE    ERROR1            ;STACK DID NOT DECREMENT CORRECTLY
002046 022767 002030 176720     3$:      CMP    #8$,STBOT-4       ;WAS PROPER PC SAVED
002054 001025                            BNE    ERROR1            ;PROPER PC WAS NOT SAVED
002056 005767 176714            4$:      TST    STBOT-2           ;WAS OLD PSW SAVED
002062 001022                            BNE    ERROR1            ;WRONG PSW SAVED
002064 062706 000004            5$:      ADD    #4,SP             ;RESET STACK POINTER
002070 012767 002112 175736              MOV    #6$,RTRAP1                ;SET UP NEW PC IN VECTOR
002076 005067 175734                     CLR    RTRAP1+2                  ;SET UP NEW PSW IN VECTOR
002102 010037 177776                     MOV    R0,@#STATUS       ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002106 104777                            TRAPC                    ;DO TRAP WITH LOWER BYTE ALL ONES
002110 000407                            BR     ERROR1            ;INSTRUCTION FAILED TO TRAP
002112 005767 175660            6$:      TST    STATUS            ;IS NEW PSW CORRECT
002116 001004                            BNE    ERROR1            ;NEW PSW WRONG
002120 020067 176652            7$:      CMP    R0,STBOT-2        ;WAS OLD STATUS STORED
002124 001001                            BNE    ERROR1            ;OLD STATUS WRONG
002126 000434                            BR     CPUHLT+34         ;GET OVER ERROR CALL TO NEXT TEST IF NO ERROR
                                ;*******************************************************************************
                                ;THIS ERROR IS USED FOR THE ENTIRE CPU,TRAPS AND EIS PORTION OF
                                ;THIS TEST
                                ;*******************************************************************************
002130 012737 000001 001002     ERROR1:  MOV    #1,@#$FATAL                      ;SET UP FATAL ERROR NUMBER
002136 012767 000001 176634              MOV    #1,$MSGTY                        ;SET FATAL ERROR FLAG
002144 032737 000001 001020              BIT    #1,@#$ENV                        ;UNDER APT
002152 001004                            BNE    CPUHLT                           ;YES, THEN DO NOT PRINT
002154 012700 002166                     MOV    #CPUMSG,R0
002160 004767 131144                     JSR    PC,TYPE                          ;TYPE MSG
002164 000777                   CPUHLT:  BR     .

002166 040506 046111 042105     CPUMSG: .ASCIZ  /FAILED DURING CPU TESTS/<12><15>
002174 042040 051125 047111
002202 020107 050103 020125
002210 042524 052123 005123
002216 000015
                                .EVEN

                                ;*******************************************************************************
                                ;TEST 4 TEST TRAP OF IOT INSTRUCTION
                                ;*******************************************************************************
002220                          TS4:
002220 012706 001000                     MOV    #STBOT,SP         ;INITIALIZE THE STACK POINTER
002224 012767 002246 175566              MOV    #1$,RTRAP2                ;SET UP NEW PC IN VECTOR
002232 010067 175564                     MOV    R0,RTRAP2+2               ;SET UP NEW PSW IN VECTOR
002236 005067 175534                     CLR    STATUS            ;CLEAR PRESENT (OLD) STATUS
002242 000004                            IOT                      ;DO TRAP
002244                          8$:
002244 000731                            BR     ERROR1            ;INSTRUCTION FAILED TO TRAP
002246 020067 175524            1$:      CMP    R0,STATUS         ;IS NEW STATUS CORRECT
002252 001326                            BNE    ERROR1            ;NEW STATUS WRONG
002254 020627 000774            2$:      CMP    SP,#STBOT-4       ;DID STACK DECREMENT CORRECTLY
002260 001323                            BNE    ERROR1            ;STACK DID NOT DECREMENT CORRECTLY
002262 022767 002244 176504     3$:      CMP    #8$,STBOT-4       ;WAS PROPER PC SAVED
002270 001317                            BNE    ERROR1            ;PROPER PC WAS NOT SAVED
```

```
002272  005767  176500          4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
002276  001314                          BNE     ERROR1          ;WRONG PSW SAVED
002300  062706  000004          5$:     ADD     #4,SP           ;RESET STACK POINTER
002304  012767  002326  175506          MOV     #6$,RTRAP2              ;SET UP NEW PC IN VECTOR
002312  005067  175504                  CLR     RTRAP2+2                ;SET UP NEW PSW IN VECTOR
002316  010037  177776                  MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002322  000004                          IOT                     ;DO TRAP
002324  000701                          BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
002326  005767  175444          6$:     TST     STATUS          ;IS NEW PSW CORRECT
002332  001276                          BNE     ERROR1          ;NEW PSW WRONG
002334  020067  176436          7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
002340  001273                          BNE     ERROR1          ;OLD STATUS WRONG
                                ;*************************************************************************
                                ;TEST 5 TEST TRAP OF EMT INSTRUCTION
                                ;*************************************************************************
002342                          TS5:
002342  012706  001000                  MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002346  012767  002370  175454          MOV     #1$,RTRAP3              ;SET UP NEW PC IN VECTOR
002354  010067  175452                  MOV     R0,RTRAP3+2             ;SET UP NEW PSW IN VECTOR
002360  005067  175412                  CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
002364  104000                          EMT                     ;DO TRAP
002366                          8$:
002366  000660                          BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
002370  020067  175402          1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
002374  001255                          BNE     ERROR1          ;NEW STATUS WRONG
002376  020627  000774          2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
002402  001252                          BNE     ERROR1          ;STACK DID NOT DECREMENT CORRECTLY
002404  022767  002366  176362  3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
002412  001246                          BNE     ERROR1          ;PROPER PC WAS NOT SAVED
002414  005767  176356          4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
002420  001243                          BNE     ERROR1          ;WRONG PSW SAVED
002422  062706  000004          5$:     ADD     #4,SP           ;RESET STACK POINTER
002426  012767  002450  175374          MOV     #6$,RTRAP3              ;SET UP NEW PC IN VECTOR
002434  005067  175372                  CLR     RTRAP3+2                ;SET UP NEW PSW IN VECTOR
002440  010037  177776                  MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002444  104377                          EMTA                    ;DO TRAP WITH LOWER BYTE ALL ONES
002446  000630                          BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
002450  005767  175322          6$:     TST     STATUS          ;IS NEW PSW CORRECT
002454  001225                          BNE     ERROR1          ;NEW PSW WRONG
002456  020067  176314          7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
002462  001222                          BNE     ERROR1          ;OLD STATUS WRONG
002464  000401                          BR      ERROR2+2        ;WE MUST GET OVER ERROR CALL AT END OF THIS TEST
                                ;*************************************************************************
                                ;THIS ERROR IS NEEDED BECAUSE BRANCHES IN TRAP TESTS BEYOND HERE CAN NOT
                                ;REACH ERROR1.
                                ;*************************************************************************
002466  000620                  ERROR2: BR      ERROR1


                                ;*************************************************************************
                                ;TEST 6 TEST TRAP OF TRACE-TRAP INSTRUCTION
                                ;*************************************************************************
002470                          TS6:
002470  012706  001000                  MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002474  012767  002516  175312          MOV     #1$,RTRAP4              ;SET UP NEW PC IN VECTOR
002502  010067  175310                  MOV     R0,RTRAP4+2             ;SET UP NEW PSW IN VECTOR
```

```
002506  005067  175264                    CLR     STATUS              ;CLEAR PRESENT (OLD) STATUS
002512  000003                            TRT                         ;DO TRAP
002514                            8$:
002514  000764                            BR      ERROR2              ;INSTRUCTION FAILED TO TRAP
002516  020067  175254            1$:     CMP     R0,STATUS           ;IS NEW STATUS CORRECT
002522  001361                            BNE     ERROR2              ;NEW STATUS WRONG
002524  020627  000774            2$:     CMP     SP,#STBOT-4         ;DID STACK DECREMENT CORRECTLY
002530  001356                            BNE     ERROR2              ;STACK DID NOT DECREMENT CORRECTLY
002532  022767  002514  176234    3$:     CMP     #8$,STBOT-4         ;WAS PROPER PC SAVED
002540  001352                            BNE     ERROR2              ;PROPER PC WAS NOT SAVED
002542  005767  176230            4$:     TST     STBOT-2             ;WAS OLD PSW SAVED
002546  001347                            BNE     ERROR2              ;WRONG PSW SAVED
002550  062706  000004            5$:     ADD     #4,SP               ;RESET STACK POINTER
002554  012767  002576  175232            MOV     #6$,RTRAP4                    ;SET UP NEW PC IN VECTOR
002562  005067  175230                    CLR     RTRAP4+2                      ;SET UP NEW PSW IN VECTOR
002566  010037  177776                    MOV     R0,@#STATUS         ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002572  000003                            TRT                         ;DO TRAP
002574  000734                            BR      ERROR2              ;INSTRUCTION FAILED TO TRAP
002576  005767  175174            6$:     TST     STATUS              ;IS NEW PSW CORRECT
002602  001331                            BNE     ERROR2              ;NEW PSW WRONG
002604  020067  176166            7$:     CMP     R0,STBOT-2          ;WAS OLD STATUS STORED
002610  001326                            BNE     ERROR2              ;OLD STATUS WRONG
                                  ;PDP-11 ILLEGAL AND ADDRESS INSTRUCTION TEST
                                  ;ALL INSTRUCTIONS THAT ARE RESERVED
                                  ;SHOULD TRAP TO LOCATION 4, AND THE
                                  ;PC THAT POINTS TO THE TRAPPING INSTRUCTION
                                  ;SHOULD BE PLACED ON THE STACK

                                  ;*****************************************************************************
                                  ;TEST 7 TEST TRAP OF ILLEGAL INSTRUCTION
                                  ;*****************************************************************************
002612                            TS7:
002612  012706  001000                    MOV     #STBOT,SP           ;INITIALIZE THE STACK POINTER
002616  012767  002640  175160            MOV     #1$,RTRAP5                    ;SET UP NEW PC IN VECTOR
002624  010067  175156                    MOV     R0,RTRAP5+2                   ;SET UP NEW PSW IN VECTOR
002630  005067  175142                    CLR     STATUS              ;CLEAR PRESENT (OLD) STATUS
002634  000100                            JMP     %0                            ;DO TRAP
002636                            8$:
002636  000713                            BR      ERROR2              ;INSTRUCTION FAILED TO TRAP
002640  020067  175132            1$:     CMP     R0,STATUS           ;IS NEW STATUS CORRECT
002644  001310                            BNE     ERROR2              ;NEW STATUS WRONG
002646  020627  000774            2$:     CMP     SP,#STBOT-4         ;DID STACK DECREMENT CORRECTLY
002652  001305                            BNE     ERROR2              ;STACK DID NOT DECREMENT CORRECTLY
002654  022767  002636  176112    3$:     CMP     #8$,STBOT-4         ;WAS PROPER PC SAVED
002662  001301                            BNE     ERROR2              ;PROPER PC WAS NOT SAVED
002664  005767  176106            4$:     TST     STBOT-2             ;WAS OLD PSW SAVED
002670  001276                            BNE     ERROR2              ;WRONG PSW SAVED
002672  062706  000004            5$:     ADD     #4,SP               ;RESET STACK POINTER
002676  012767  002720  175100            MOV     #6$,RTRAP5                    ;SET UP NEW PC IN VECTOR
002704  005067  175076                    CLR     RTRAP5+2                      ;SET UP NEW PSW IN VECTOR
002710  010037  177776                    MOV     R0,@#STATUS         ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002714  000100                            JMP     %0                            ;DO TRAP
002716  000663                            BR      ERROR2              ;INSTRUCTION FAILED TO TRAP
002720  005767  175052            6$:     TST     STATUS              ;IS NEW PSW CORRECT
002724  001260                            BNE     ERROR2              ;NEW PSW WRONG
002726  020067  176044            7$:     CMP     R0,STBOT-2          ;WAS OLD STATUS STORED
```

```
002732  001255                            BNE     ERROR2          ;OLD STATUS WRONG
                                  ;*******************************************************************************
                                  ;TEST 10         TEST TRAP OF ALL ILLEGAL INSTRUCTION
                                  ;*******************************************************************************
002734                            TS10:
002734  012706  001000                    MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002740  012767  002762  175036            MOV     #1$,RTRAP5              ;SET UP NEW PC IN VECTOR
002746  010067  175034                    MOV     R0,RTRAP5+2             ;SET UP NEW PSW IN VECTOR
002752  005067  175020                    CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
002756  004000                            JSR     %0,%0                   ;DO TRAP
002760                            8$:
002760  000642                            BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
002762  020067  175010            1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
002766  001237                            BNE     ERROR2          ;NEW STATUS WRONG
002770  020627  000774            2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
002774  001234                            BNE     ERROR2          ;STACK DID NOT DECREMENT CORRECTLY
002776  022767  002760  175770    3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
003004  001230                            BNE     ERROR2          ;PROPER PC WAS NOT SAVED
003006  005767  175764            4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
003012  001225                            BNE     ERROR2          ;WRONG PSW SAVED
003014  062706  000004            5$:     ADD     #4,SP           ;RESET STACK POINTER
003020  012767  003042  174756            MOV     #6$,RTRAP5              ;SET UP NEW PC IN VECTOR
003026  005067  174754                    CLR     RTRAP5+2                ;SET UP NEW PSW IN VECTOR
003032  010037  177776                    MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
003036  004000                            JSR     %0,%0                   ;DO TRAP
003040  000612                            BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
003042  005767  174730            6$:     TST     STATUS          ;IS NEW PSW CORRECT
003046  001207                            BNE     ERROR2          ;NEW PSW WRONG
003050  020067  175722            7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
003054  001204                            BNE     ERROR2          ;OLD STATUS WRONG

                                  ;*******************************************************************************
                                  .SBTTL  DATA PATH TESTS
                                  ;
                                  ;       THE DATA PATH TESTS ARE USED TO VERIFY THAT VARIOUS
                                  ;DATA PATTERNS CAN BE SUCCESSFULLY MOVED THROUGH THE DATA PATHS
                                  ;MOVE AND COMPARE MODE 2,3 INSTRUCTIONS ARE USED TO PASS AND
                                  ;TEST VARIOUS DATA PATTERNS IN THE DATA PATHS.
                                  ;       THE TEST EXERCISES THE INTERNAL DATA PATHS, AND THE UNIBUS
                                  ;DATA TRANSCIEVERS.
                                  ;       IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0)
                                  ;TO SEE WHICH BITS OF THE DATA PATH ARE FAILING.
003056  012737  002130  000030            MOV     #ERROR1,@#30    ;SET UP VECTOR FOR ERROR CALLS
003064  012737  000340  000032            MOV     #340,@#32       ;SET UP NEW PSW
003072  012737  021336  000004            MOV     #T04,@#4        ;SET UP FOR UNEXPECTED TRAP TO 4
003100  012737  021340  000010            MOV     #T010,@#10      ;SET UP FOR UNEXPECTED TRAP TO 10
003106  012737  021342  000014            MOV     #T014,@#14      ;SET UP FOR UNEXPECTED TRAP TO 14
003114  012737  021350  000034            MOV     #T034,@#34      ;SET UP FOR UNEXPECTED TRAP TO 34


                                  ;*******************************************************************************
                                  ;TEST 11         TEST OF ZEROES IN THE DATA PATH
                                  ;*******************************************************************************
003122                            TS11:
003122  012737  000000  000000            MOV     #0,@#0          ;MOVE ZEROES THRU ADDRESS LINES, DATA
                                                                  ;LINES AND INTERNAL PATHS
```

```
003130  005737  000000                         TST     @#0             ;SUCCESSFUL?
003134  001401                                 BEQ     TS12
003136  104000                                 EMT                     ;DATA INCORRECT

                                       ;**********************************************************************
                                       ;TEST 12         TEST OF PATTERN 125252 IN DATA PATH
                                       ;**********************************************************************
003140                                 TS12:
003140  012737  125252  000000                 MOV     #125252,@#0     ;MOVE ALTERNATING ONES AND ZEROES
                                                                       ;THRU DATA PATHS
003146  022737  125252  000000                 CMP     #125252,@#0     ;SUCCESSFUL
003154  001401                                 BEQ     TS13
003156  104000                                 EMT                     ;DATA INCORRECT

                                       ;**********************************************************************
                                       ;TEST 13         TEST OF PATTERN 052525 IN DATA PATH
                                       ;**********************************************************************
003160                                 TS13:
003160  012737  052525  000000                 MOV     #052525,@#0     ;MOVE ALTERNATING ZEROES AND ONES
                                                                       ;THRU DATA PATH
003166  022737  052525  000000                 CMP     #052525,@#0     ;SUCCESSFUL?
003174  001401                                 BEQ     TS14
003176  104000                                 EMT                     ;DATA INCORRECT

                                       ;**********************************************************************
                                       ;TEST 14         TEST OF ALL ONES IN DATA PATH
                                       ;**********************************************************************
003200                                 TS14:
003200  012737  177777  000000                 MOV     #177777,@#0     ;MOVE ONES THRU DATA PATH
003206  022737  177777  000000                 CMP     #177777,@#0     ;SUCCESSFUL
003214  001401                                 BEQ     TS15
003216  104000                                 EMT                     ;DATA INCORRECT

                                       ;**********************************************************************
                                       .SBTTL  B-REGISTER TEST
                                       ;
                                       ;        THE B-REGISTER (LOCATION 0) SHIFTING LOGIC TESTS ARE USED
                                       ;TO TEST THAT THE B-REGISTER CAN HOLD VARIOUS DATA PATTERNS AND THAT
                                       ;THE ASSOCIATED LOGIC SUPPORTS THE SHIFTING FUNCTIONS WITHIN THE
                                       ;B-REGISTER AND C-BIT.
                                       ;A ONE IS SHIFTED THROUGH EVERY BIT IN THE B-REGISTER AND C-BIT IN
                                       ;BOTH DIRECTIONS.
                                       ;        THE B-REGISTER ITSELF IS TESTED IN ITS ABILITY AS A BUFFER AND AS
                                       ;A SHIFT REGISTER.  DATA IS ALSO PASSED THROUGH THE DATA PATH AND ALU,
                                       ;        IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0) TO SEE
                                       ;WHICH BITS OF THE B-REGISTER MAY BE FAILING.
                                       ;
                                       ;**********************************************************************
                                       ;TEST 15         SHIFT BIT 0 TO BIT 1
                                       ;**********************************************************************
003220                                 TS15:
003220  000241                                 CLC                     ;CLEAR CARRY BIT
003222  012737  000001  000000                 MOV     #1,@#0          ;LOAD A 1
003230  006137  000000                         ROL     @#0             ;SHIFT LEFT
003234  022737  000002  000000                 CMP     #2,@#0          ;SUCCESSFUL
003242  001401                                 BEQ     TS16
```

```
003244  104000                          EMT                             ;BIT 1 NOT SET

                                ;*******************************************************************************
                                ;TEST 16        SHIFT CARRY INTO BIT 0
                                ;*******************************************************************************
003246                          TS16:
003246  012737  000000  000000          MOV     #0,@#0                  ;CLEAR LOCATION
003254  000261                          SEC                             ;SET CARRY
003256  006137  000000                  ROL     @#0                     ;ROTATE CARRY BIT TO BIT 0
003262  103001                          BCC     CARRY1
003264  104000                          EMT                             ;CARRY CLEAR
003266  022737  000001  000000  CARRY1: CMP     #1,@#0                  ;BIT 0 SET
003274  001401                          BEQ     TS17
003276  104000                          EMT                             ;BIT 0 NOT SET

                                ;*******************************************************************************
                                ;TEST 17        LEFT SHIFT FROM BIT 0 TO C-BIT
                                ;*******************************************************************************
003300                          TS17:
003300  012737  000001  000000          MOV     #1,@#0                  ;SET BIT 0
003306  012700  177757                  MOV     #-21,R0                 ;SET BIT COUNTER
003312  000241                          CLC                             ;CLEAR C-BIT
003314  005200                  SHL:    INC     R0                      ;INCREMENT BIT COUNTER
003316  001404                          BEQ     SHLE                    ;BR TO ERROR HALT IF BIT IS LOST
003320  006137  000000                  ROL     @#0                     ;SHIFT LEFT ONE POSITION
003324  103373                          BCC     SHL                     ;BRANCH IF C-BIT NOT SET
003326  001401                          BEQ     TS20
003330                          SHLE:
003330  104000                          EMT                             ;LEFT SHIFTING LOGIC FAILED

                                ;*******************************************************************************
                                ;TEST 20        SHIFT BIT 15 TO BIT 14
                                ;*******************************************************************************
003332                          TS20:
003332  012737  100000  000000          MOV     #100000,@#0             ;SET BIT 15
003340  000241                          CLC                             ;CLEAR CARRY
003342  006037  000000                  ROR     @#0                     ;SHIFT BIT 15 TO BIT 14
003346  022737  040000  000000          CMP     #40000,@#0              ;SUCCESSFUL
003354  001401                          BEQ     TS21
003356  104000                          EMT                             ;BIT 14 NOT SET

                                ;*******************************************************************************
                                ;TEST 21        RIGHT SHIFT FROM BIT 15 TO C-BIT
                                ;*******************************************************************************
003360                          TS21:
003360  012737  100000  000000          MOV     #100000,@#0             ;SET BIT 15
003366  012700  177757                  MOV     #-21,R0                 ;SET BIT COUNTER
003372  000241                          CLC                             ;CLEAR C-BIT
003374  005200                  SHR:    INC     R0                      ;INCREMENT BIT COUNTER
003376  001404                          BEQ     SHRE                    ;BR TO ERROR HALT IF BIT IS LOST
003400  006037  000000                  ROR     @#0                     ;ROTATE RIGHT ONE POSITION
003404  103373                          BCC     SHR                     ;BRANCH IF C-BIT CLEAR
003406  001401                          BEQ     TS22
003410                          SHRE:
003410  104000                          EMT                             ;RIGHT SHIFT LOGIC FAILED
```

```
                                  ;*******************************************************************************
                                  .SBTTL  SCRATCH PAD TESTS
                                  ;
                                  ;       THE SCRATCH PAD TESTS ARE USED TO VERIFY THAT VARIOUS
                                  ;DATA PATTERNS CAN BE SUCCESSFULLY HELD IN THE SCRATCH PAD
                                  ;CIRCUITRY.  MOVE AND COMPARE INSTRUCTIONS ARE USED TO TEST THAT
                                  ;RO CAN HOLD VARIOUS DATA PATTERNS.  EACH DATA PATTERN IS
                                  ;MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR SCOPING.  THE
                                  ;SUCCESSFUL COMPLETION OF THESE TESTS SHOULD VERIFY THE CIRCUITRY EXTERNAL
                                  ;TO THE SCRATCH PAD ITSELF.
                                  ;       THE REMAINDER OF THE GENERAL REGISTERS ARE TESTED BY MOVING
                                  ;A BIT INTO BIT 0 OF THE REGISTER AND SHIFTING IT LEFT ONE
                                  ;BIT AT A TIME INTO THE CARRY BIT.  THE RESULT IS THEN CHECKED TO INSURE THAT
                                  ;NO BITS WERE PICKED. THE PROCEDURE IS THEN REPEATED UNDER OPPOSITE
                                  ;CONDITIONS. THE GENERAL REGISTER AND THE CARRY BIT ARE SET TO
                                  ;ALL ONES, AND A ZERO IS SHIFTED LEFT FROM BIT 0 INTO THE CARRY BIT.
                                  ;THE RESULT IS THEN CHECKED TO INSURE THAT NO ZEROES WERE PICKED.
                                  ;       AT THIS POINT ALL OF THE GENERAL REGISTERS HAVE BEEN EXERCISED
                                  ;AS WELL AS REGISTER 11.
                                  ;
                                  ;*******************************************************************************
                                  ;TEST 22          TEST IF RO CAN HOLD ALL ZEROES
                                  ;*******************************************************************************
003412                            TS22:

003412  012700  000000                    MOV     #0,RO           ;MOVE ZEROES TO RO
003416  005700                            TST     RO              ;SUCCESSFUL?
003420  001401                            BEQ     TS23
003422  104000                            EMT                     ;RO NOT 0

                                  ;*******************************************************************************
                                  ;TEST 23          TEST IF RO CAN HOLD ONES AND ZEROES
                                  ;*******************************************************************************
003424                            TS23:
003424  012700  125252                    MOV     #125252,RO      ;MOVE ALTERNATING ONES AND ZEROES TO RO
003430  020027  125252                    CMP     RO,#125252      ;SUCCESSFUL?
003434  001401                            BEQ     TS24
003436  104000                            EMT                     ;RO NOT 125252

                                  ;*******************************************************************************
                                  ;TEST 24          TEST IF RO CAN HOLD ZEROES AND ONES
                                  ;*******************************************************************************
003440                            TS24:
003440  012700  052525                    MOV     #052525,RO      ;MOVE ALTERNATING ZEROES AND ONES TO RO
003444  020027  052525                    CMP     RO,#052525      ;SUCCESSFUL?
003450  001401                            BEQ     TS25
003452  104000                            EMT                     ;RO NOT 52525

                                  ;*******************************************************************************
                                  ;TEST 25          TEST IF RO CAN HOLD ALL ONES
                                  ;*******************************************************************************
003454                            TS25:
003454  012700  177777                    MOV     #177777,RO      ;MOVE ALL ONES TO RO
003460  020027  177777                    CMP     RO,#177777      ;SUCCESSFUL?
003464  001401                            BEQ     TS26
003466  104000                            EMT                     ;RO NOT 177777
```

```

                                    ;*******************************************************************************
                                    ;TEST 26          TEST IF R1 CAN HOLD A ONE IN ALL BITS
                                    ;*******************************************************************************
003470                              TS26:
003470  012701  000001                      MOV     #1,R1                   ;SET BIT 0
003474  012700  177757                      MOV     #-21,R0                 ;SET BIT COUNTER
003500  000241                              CLC                             ;CLEAR C-BIT
003502  005200                      REG1:   INC     R0                      ;INCREMENT BIT COUNTER
003504  001403                              BEQ     REG1E                   ;BR TO ERROR HALT IF BIT IS LOST
003506  006101                              ROL     R1                      ;ROTATE 1 POSITION
003510  103374                              BCC     REG1                    ;ALL DONE
003512  001401                              BEQ     TS27
003514                              REG1E:
003514  104000                              EMT                             ;FAILURE WITH R1

                                    ;*******************************************************************************
                                    ;TEST 27          TEST IF R1 CAN HOLD A ZERO IN ALL BITS
                                    ;*******************************************************************************
003516                              TS27:
003516  012701  177776                      MOV     #-2,R1                  ;SET ALL ONES IN R1 EXCEPT FOR BIT 0
003522  012700  177757                      MOV     #-21,R0                 ;SET BIT COUNTER
003526  000261                              SEC                             ;SET C-BIT
003530  005200                      REG1A:  INC     R0                      ;INCREMENT COUNTER
003532  001405                              BEQ     R1ERR                   ;BR TO ERROR HALT IF COUNTER=0
003534  006101                              ROL     R1                      ;ROTATE 1 POSITION
003536  103774                              BCS     REG1A                   ;CONTINUE UNTIL C-BIT IS CLEAR
003540  022701  177777                      CMP     #-1,R1                  ;CHECK DATA IN R1
003544  001401                              BEQ     TS30
003546                              R1ERR:
003546  104000                              EMT                             ;FAILURE WITH R1
                                    ;*******************************************************************************
                                    ;TEST 30          TEST IF R2 CAN HOLD A ONE IN ALL BITS
                                    ;*******************************************************************************
003550                              TS30:
003550  012702  000001                      MOV     #1,R2                   ;SET BIT 0
003554  012700  177757                      MOV     #-21,R0                 ;SET BIT COUNTER
003560  000241                              CLC                             ;CLEAR C-BIT
003562  005200                      REG2:   INC     R0                      ;INCREMENT BIT COUNTER
003564  001403                              BEQ     REG2E                   ;BR TO ERROR HALT IF BIT IS LOST
003566  006102                              ROL     R2                      ;ROTATE 1 POSITION
003570  103374                              BCC     REG2                    ;ALL DONE
003572  001401                              BEQ     TS31
003574                              REG2E:
003574  104000                              EMT                             ;FAILURE WITH R2

                                    ;*******************************************************************************
                                    ;TEST 31          TEST IF R2 CAN HOLD A ZERO IN ALL BITS
                                    ;*******************************************************************************
003576                              TS31:
003576  012702  177776                      MOV     #-2,R2                  ;SET ALL ONES IN R2 EXCEPT FOR BIT 0
003602  012700  177757                      MOV     #-21,R0                 ;SET BIT COUNTER
003606  000261                              SEC                             ;SET C-BIT
003610  005200                      REG2B:  INC     R0                      ;INCREMENT BIT COUNTER
003612  001405                              BEQ     R2ERR                   ;BR TO ERROR HALT IF COUNTER=0
003614  006102                              ROL     R2                      ;ROTATE 1 POSITION
```

```
003616  103774                    BCS     REG2B           ;CONTINUE UNTIL C-BIT IS CLEAR
003620  022702  177777            CMP     #-1,R2          ;CHECK DATA IN R2
003624  001401                    BEQ     TS32
003626                    R2ERR:
003626  104000                    EMT                     ;FAILURE WITH R2

                          ;*******************************************************************************
                          ;TEST 32         TEST IF R3 CAN HOLD A ONE IN ALL BITS
                          ;*******************************************************************************
003630                    TS32:
003630  012703  000001            MOV     #1,R3           ;SET BIT 0
003634  012700  177757            MOV     #-21,R0         ;SET BIT COUNTER
003640  000241                    CLC                     ;CLEAR C-BIT
003642  005200            REG3:   INC     R0              ;INCREMENT BIT COUNTER
003644  001403                    BEQ     REG3E           ;BR TO ERROR HALT IF BIT IS LOST
003646  006103                    ROL     R3              ;ROTATE 1 POSITION
003650  103374                    BCC     REG3            ;ALL DONE
003652  001401                    BEQ     TS33
003654                    REG3E:
003654  104000                    EMT                     ;FAILURE WITH R3

                          ;*******************************************************************************
                          ;TEST 33         TEST IF R3 CAN HOLD A ZERO IN ALL BITS
                          ;*******************************************************************************
003656                    TS33:
003656  012703  177776            MOV     #-2,R3          ;SET ALL ONES IN R3 EXCEPT FOR BIT 0
003662  012700  177757            MOV     #-21,R0         ;SET BIT COUNTER
003666  000261                    SEC                     ;SET C-BIT
003670  005200            REG3A:  INC     R0              ;INCREMENT BIT COUNTER
003672  001405                    BEQ     R3ERR           ;BR TO ERROR HALT IF COUNTER=0
003674  006103                    ROL     R3              ;ROTATE 1 POSITION
003676  103774                    BCS     REG3A           ;CONTINUE UNTIL C-BIT IS CLEAR
003700  022703  177777            CMP     #-1,R3          ;CHECK DATA
003704  001401                    BEQ     TS34
003706                    R3ERR:
003706  104000                    EMT                     ;FAILURE WITH R3

                          ;*******************************************************************************
                          ;TEST 34         TEST IF R4 CAN HOLD A ONE IN ALL BITS
                          ;*******************************************************************************
003710                    TS34:
003710  012704  000001            MOV     #1,R4           ;SET BIT 0
003714  012700  177757            MOV     #-21,R0         ;SET BIT COUNTER
003720  000241                    CLC                     ;CLEAR C-BIT
003722  005200            REG4:   INC     R0              ;INCREMENT BIT COUNTER
003724  001403                    BEQ     REG4E           ;BR TO ERROR HALT IF BIT IS LOST
003726  006104                    ROL     R4              ;ROTATE 1 POSITION
003730  103374                    BCC     REG4            ;ALL DONE
003732  001401                    BEQ     TS35
003734                    REG4E:
003734  104000                    EMT                     ;FAILURE WITH R4

                          ;*******************************************************************************
                          ;TEST 35         TEST IF R4 CAN HOLD A ZERO IN ALL BITS
                          ;*******************************************************************************
003736                    TS35:
```

```
003736  012704  177776                  MOV     #-2,R4          ;SET ALL ONES IN R4 EXCEPT FOR BIT 0
003742  012700  177757                  MOV     #-21,R0         ;SET BIT COUNTER
003746  000261                          SEC                     ;SET C-BIT
003750  005200                  REG4A:  INC     R0              ;INCREMENT BIT COUNTER
003752  001405                          BEQ     R4ERR           ;BR TO ERROR HALT IF COUNTER=0
003754  006104                          ROL     R4              ;ROTATE 1 POSITION
003756  103774                          BCS     REG4A           ;CONTINUE UNTIL C-BIT IS CLEAR
003760  022704  177777                  CMP     #-1,R4          ;CHECK DATA
003764  001401                          BEQ     TS36
003766                          R4ERR:
003766  104000                          EMT                     ;FAILURE WITH R4


                                ;*******************************************************************************
                                ;TEST 36        TEST IF R5 CAN HOLD A ONE IN ALL BITS
                                ;*******************************************************************************
003770                          TS36:
003770  012705  000001                  MOV     #1,R5           ;SET BIT 0
003774  012700  177757                  MOV     #-21,R0         ;SET BIT COUNTER
004000  000241                          CLC                     ;CLEAR C-BIT
004002  005200                  REG5:   INC     R0              ;INCREMENT BIT COUNTER
004004  001403                          BEQ     REG5E           ;BR TO ERROR HALT IF BIT IS LOST
004006  006105                          ROL     R5              ;ROTATE 1 POSITION
004010  103374                          BCC     REG5            ;ALL DONE
004012  001401                          BEQ     TS37
004014                          REG5E:
004014  104000                          EMT                     ;FAILURE WITH R5

                                ;*******************************************************************************
                                ;TEST 37        TEST IF R5 CAN HOLD A ZERO IN ALL BITS
                                ;*******************************************************************************
004016                          TS37:
004016  012705  177776                  MOV     #-2,R5          ;SET ALL ONES IN R5 EXCEPT FOR BIT 0
004022  012700  177757                  MOV     #-21,R0         ;SET BIT COUNTER
004026  000261                          SEC                     ;SET C-BIT
004030  005200                  REG5A:  INC     R0              ;INCREMENT BIT COUNTER
004032  001405                          BEQ     R5ERR           ;BR TO ERROR HALT IF COUNTER=0
004034  006105                          ROL     R5              ;ROTATE 1 POSITION
004036  103774                          BCS     REG5A           ;CONTINUE UNTIL C-BIT IS C;EAR
004040  022705  177777                  CMP     #-1,R5          ;CHECK DATA
004044  001401                          BEQ     TS40
004046                          R5ERR:
004046  104000                          EMT                     ;FAILURE WITH R5

                                ;*******************************************************************************
                                ;TEST 40        TEST IF R6 CAN HOLD A ONE IN ALL BITS
                                ;*******************************************************************************
004050                          TS40:
004050  012706  000001                  MOV     #1,R6           ;SET BIT 0
004054  012700  177757                  MOV     #-21,R0         ;SET BIT COUNTER
004060  000241                          CLC                     ;CLEAR C-BIT
004062  005200                  REG6:   INC     R0              ;INCREMENT BIT COUNTER
004064  001403                          BEQ     REG6E           ;BR TO ERROR HALT IF BIT IS LOST
004066  006106                          ROL     R6              ;ROTATE 1 POSITION
004070  103374                          BCC     REG6            ;ALL DONE
004072  001401                          BEQ     TS41
```

```
004074                           REG6E:
004074  104000                           EMT                        ;FAILURE WITH R6

                                 ;*******************************************************************************
                                 ;TEST 41         TEST IF R6 CAN HOLD A ZERO IN ALL BITS
                                 ;*******************************************************************************
004076                           TS41:
004076  012706  177776                   MOV     #-2,R6             ;SET ALL ONES IN R6 EXCEPT FOR BIT 0
004102  012700  177757                   MOV     #-21,R0            ;SET BIT COUNTER
004106  000261                           SEC                        ;SET C-BIT
004110  005200                   REG6A:  INC     R0                 ;INCREMENT BIT COUNT
004112  001405                           BEQ     R6ERR              ;BR TO ERROR HALT IF COUNTER=0
004114  006106                           ROL     R6                 ;ROTATE 1 POSITION
004116  103774                           BCS     REG6A              ;CONTINUE UNTIL C-BIT IS CLEAR
004120  022706  177777                   CMP     #-1,R6             ;CHECK DATA
004124  001401                           BEQ     TS42
004126                           R6ERR:
004126  104000                           EMT                        ;FAILURE WITH R6

                                 ;*******************************************************************************
                                 .SBTTL  PSW TESTS
                                 ;
                                 ;       THE PSW TESTS ARE USED TO VERIFY THAT VARIOUS DATA
                                 ;PATTERNS CAN BE SUCCESSFULLY HELD IN THE PSW AND THAT THE
                                 ;PSW ADDRESSING LOGIC IS FUNCTIONING.  MOVE AND COMPARE INSTRUCTIONS
                                 ;ARE USED TO TEST THAT THE PSW CAN HOLD VARIOUS DATA PATTERNS.
                                 ;EACH DATA PATTERN IS MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR
                                 ;SCOPING.
                                 ;       THE PSW REGISTER IS TESTED, THE CC INPUTS ARE TESTED
                                 ;LATER IN THE MICROCODE TESTS.  SETTING OF THE T-BIT BY THE
                                 ;TEST PATTERNS IS PURPOSELY AVOIDED. TESTING OF THE
                                 ;T-BIT TRAP CIRCUITRY IS LEFT FOR THE TRAP TEST.
                                 ;
                                 ;*******************************************************************************
                                 ;TEST 42         TEST IF PSW WILL HOLD ZEROES
                                 ;*******************************************************************************
004130                           TS42:
004130  012706  001000                   MOV     #STBOT,R6
004134  012737  000000  177776           MOV     #0,@#PS            ;SET PSW TO ZERO
004142  005737  177776                   TST     @#PS               ;SUCCESSFUL
004146  001401                           BEQ     TS43
004150  104000                           EMT                        ;PSW NOT 0

                                 ;*******************************************************************************
                                 ;TEST 43         TEST IF PSW WILL HOLD ONES AND ZEROES
                                 ;*******************************************************************************
004152                           TS43:
004152  012737  000252  177776           MOV     #252,@#PS          ;MOVE ALT. ONES AND ZEROES TO PSW
004160  023727  177776  000252           CMP     @#PS,#252          ;SUCCESSFUL?
004166  001401                           BEQ     TS44
004170  104000                           EMT                        ;PSW NOT 252

                                 ;*******************************************************************************
                                 ;TEST 44         TEST IF PSW (EXCEPT T-BIT) WILL HOLD ZEROES AND ONES
                                 ;*******************************************************************************
004172                           TS44:
```

```
004172  012737  000105  177776          MOV     #105,@#PS       ;MOVE ALT. ONES AND ZEROES TO PSW
004200  023727  177776  000105          CMP     @#PS,#105       ;SUCCESSFUL?
004206  001401                          BEQ     TS45
004210  104000                          EMT                     ;PSW NOT 105

                                ;*******************************************************************************
                                ;TEST 45        TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES
                                ;*******************************************************************************
004212                          TS45:
004212  012737  000357  177776          MOV     #357,@#PS       ;MOVE ONES TO PSW
004220  023727  177776  000357          CMP     @#PS,#357       ;SUCCESSFUL
004226  001401                          BEQ     TS46
004230  104000                          EMT                     ;PSW NOT 357
                                ;*******************************************************************************
                                .SBTTL  MICROCODE TESTS

                                ;
                                ;       THE TEST EXERCISES  BRANCHES IN THE MICROCODE BY
                                ;TESTING AT LEAST ONE INSTRUCTION FROM EVERY CLASS OF INSTRUCTION IN
                                ;ALL POSSIBLE MODES.  FOR EXAMPLE, TO TEST THE SINGLE OPERAND INSTRUCTIONS,
                                ;AT LEAST ONE SINGLE OPERAND INSTRUCTION IS VERIFIED IN ALL UNIQUE
                                ;ADDRESSING MODES.  BYTE MODES ARE ALSO TESTED.  AS EACH NEW
                                ;MODE IS INTRODUCED THE SAME INSTRUCTION IS TRIED AND TESTED IN
                                ;A SMALL LOOP CONVENIENT FOR SCOPING.  THE TEST IS SET UP USING
                                ;ONLY INSTRUCTIONS AND ADDRESSING MODES WHICH HAVE BEEN PREVIOUSLY
                                ;VERIFIED.
                                ;       IF THESE TESTS FAIL, CHECK THE RESULTS FOR A CLUE TO THE
                                ;FAULT.
                                ;
                                ;*******************************************************************************



                                ;*******************************************************************************
                                ;
                                ;       THE CLR INSTRUCTION IS USED TO INTRODUCE EACH ADDRESSING
                                ;MODE WITH THE SINGLE OPERAND INSTRUCTION.  FOLLOWING THE SEQUENCE CHECK,
                                ;THE CLR INSTRUCTION IS EXECUTED AND A BRANCH TEST IS EXECUTED WHICH
                                ;CHECKS THAT THE Z-BIT WAS PROPERLY SET.  THIS TEST CAN CHECK IR DECODE
                                ;AND MICROCODE FOR SOP INSTRUCTIONS WITH MODE 0.  FOLLOWING THIS TEST
                                ;SEVERAL OTHER SOP INSTRUCTIONS ARE INTRODUCED WITH MODE 0.  THESE
                                ;INSTRUCTIONS MAINPULATE DATA AND SERVE TO CHECK THE DATA RESULTS
                                ;OF THE SOP INSTRUCTIONS IN THIS TEST.  THE DATA IN THIS TEST IS
                                ;OPERATED ON BY EACH INSTRUCTION WITHOUT REINITIALIZING.
                                ;
                                ;*******************************************************************************
                                ;TEST 46        TEST MODE 0 USING SOP INST.
                                ;*******************************************************************************
004232                          TS46:
004232  005000                          CLR     R0              ;TRY THE CLEAR INST.
004234  001401                          BEQ     SOP0A
004236  104000                          EMT                     ;CLR DID NOT SET Z-BIT
004240  005200                  SOP0A:  INC     R0              ;TRY THE INCREMENT INST.
004242  005100                          COM     R0              ;TRY COMPLEMENT
004244  005200                          INC     R0
004246  100401                          BMI     SOP0B
004250  104000                          EMT                     ;NEGATE DID NOT SET N-BIT
```

```
004252  005100          SOP0B:  COM     R0                      ;TRY COMPLEMENT INST.
004254  001401                  BEQ     TS47
004256  104000                  EMT                             ;CUMMULATIVE RESULT OF CLR,INC,NEG AND COM INSTS. FAILED


                        ;*******************************************************************************
                        ;
                        ;       THIS TEST INTRODUCES THE REMAINING SOP INSTRUCIONS AND TESTS
                        ;THEM IN MODE 0.  THE PURPOSE IS TO PROVIDE A BASELINE OF
                        ;INSTRUCTIONS FOR USE IN THE SUBSEQUENT TESTS.  SINCE THE MICROCODE FOR
                        ;THESE INSTRUCTIONS IS IDENTICAL TO THAT ALREADY TESTED, ANY TROUBLE
                        ;SHOOTING EFFORTS SHOULD BE AIMED AT THE ACTUAL IR DECODE AND ALU
                        ;FUNCTIONING.
                        ;
                        ;*******************************************************************************
                        ;TEST 47        TEST REMAINDER OF SOP INSTS IN MODE 0
                        ;*******************************************************************************
004260                  TS47:
004260  005000                  CLR     R0                      ;INITIALIZE
004262  005300                  DEC     R0                      ;TRY DECREMENT INST.
004264  100401                  BMI     SOP0C
004266  104000                  EMT                             ;N-BIT NOT SET ON DEC
004270  000261          SOP0C:  SEC                             ;INITIALIZE CARRY
004272  005500                  ADC     R0                      ;TRY ADD CARRY INST
004274  001007                  BNE     SOP0D
004276  000261                  SEC                             ;INITIALIZE CARRY
004300  005600                  SBC     R0                      ;TRY SUBTRACT-CARRY INST
004302  100004                  BPL     SOP0D
004304  005100                  COM     R0
004306  005200                  INC     R0
004310  005300                  DEC     R0
004312  001401                  BEQ     TS50
004314                  SOP0D:
004314  104000                  EMT                             ; CUMMULATIVE RESULT OF ADC,SBC,COM,INC AND DEC INSTS. F

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST INTRODUCES THE BYTE CONTROL LOGIC OF THE PROCESSOR.
                        ;THE MODE 0 BYTE MICROCODE IS TESTED.  THE METHOD AND SEQUENCE
                        ;OF TESTING IS THE SAME AS THAT USED IN THE SOP MODE 0 TESTS.
                        ;
                        ;*******************************************************************************
                        ;TEST 50        TEST MODE 0 EVEN BYTE USING SOP INST
                        ;*******************************************************************************
004316                  TS50:
004316  105000                  CLRB    R0                      ;TRY CLEARING EVEN BYTE OF REGISTER
004320  001401                  BEQ     SOPB0A
004322  104000                  EMT                             ;CLRB DID NOT SET Z-BIT
004324  105100          SOPB0A: COMB    R0                      ;TRY SETTING EVEN BYTE OF REGISTER
004326  100002                  BPL     SOPB0B
004330  105200                  INCB    R0                      ;TRY INCREMENTING EVEN BYTE OF REGISTER>>
004332  001401                  BEQ     TS51
004334                  SOPB0B:
004334  104000                  EMT                             ;TEST CUMMULATIVE RESULT OF ABOVE BYTE INST.

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

```
                        ;
                        ;       THIS TEST USES THE CLR INSTRUCTION TO INTRODUCE AND TEST
                        ;SINGLE OPERAND MODE 1 INSTRUCTIONS.  AGAIN, THE CLR INSTRUCTION
                        ;IS USED TO INTRODUCE THE MICROCODE AND TO TEST THAT THE PROPER
                        ;CONDITION CODES ARE SET.  OTHER SOP INSTRUCTIONS ARE USED TO MANIPULATE
                        ;COMMON DATA TO VERIFY THAT THE CORRECT DATA IS PRODUCED.
                        ;
                        ;*************************************************************************
                        ;TEST 51        TEST MODE 1 USING SOP INST.
                        ;*************************************************************************
004336                  TS51:
004336  005000                  CLR     R0                      ;INITIALIZE R0
004340  005010                  CLR     (R0)                    ;TRY CLEAR INST W/MODE 1
004342  001401                  BEQ     SOP1A
004344  104000                  EMT                             ;CLR DID NOT SET Z-BIT
004346  005310          SOP1A:  DEC     (R0)                    ;TRY DECREMENT INST W/MODE 1
004350  100003                  BPL     SOP1B
004352  000261                  SEC                             ;INITIALIZE CARRY
004354  005510                  ADC     (R0)                    ;TRY ADD-CARRY W/MODE 1
004356  001401                  BEQ     TS52
004360                  SOP1B:
004360  104000                  EMT                             ;TEST CUMMULATIVE RESULT OF ABOVE INST

                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THE BYTE INSTRUCTION MICROCODE FOR MODE 1
                        ;SINGLE OPERAND INSTRUCTIONS.
                        ;       THIS IS THE FIRST PLACE THE SIGN EXTEND LOGIC IS EXERCISED
                        ;AND VERIFIED.
                        ;
                        ;*************************************************************************
                        ;TEST 52        TEST MODE 1 EVEN BYTE USING SOP INST
                        ;*************************************************************************
004362                  TS52:
004362  005000                  CLR     R0                      ;INITIALIZE R0
004364  005010                  CLR     (R0)                    ;INITIALIZE LOC. 0
004366  005110                  COM     (R0)
004370  105010                  CLRB    (R0)                    ;TRY TO CLEAR BYTE 0
004372  001401                  BEQ     SOPB1A
004374  104000                  EMT                             ;CLRB DID NOT SET Z-BIT
004376  005210          SOPB1A: INC     (R0)                    ;INCREMENT TO TEST WORD
004400  100005                  BPL     SOPB1B
004402  105110                  COMB    (R0)                    ;COMPLEMENT:  ODD BYTE = 376
004404  105210                  INCB    (R0)                    ;INC:  ODD BYTE = 377
004406  100002                  BPL     SOPB1B
004410  105210                  INCB    (R0)                    ;INCREMENT ODD BYTE=0
004412  001401                  BEQ     TS53
004414                  SOPB1B:
004414  104000                  EMT                             ;CHECK CUMMULATIVE RESULT OF ABOVE INST


                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THAT SINGLE OPERAND BYTE INSTRUCTIONS WILL
                        ;FUNCTION CORRECTLY FOR ODD BYTES.
                        ;       THIS IS THE FIRST TIME THAT ADDRESS LINE 0 HAS BEEN
```

```
                          ;EXERCISED.  CHECKS ARE MADE THAT THE PROPER BYTE IS MODIFIED AND
                          ;THE CONDITION CODES ARE CHECKED.  IT IS ALSO VERIFIED THAT THE UNADDRESSED
                          ;BYTE IS NOT ALTERED BY THE INSTRUCTION.
                          ;
                          ;*****************************************************************************
                          ;TEST 53         TEST MODE 1 ODD BYTE USING SOP INST
                          ;*****************************************************************************
004416                    TS53:
004416  005000                    CLR     R0                      ;INITIALIZE R0
004420  005010                    CLR     (R0)                    ;INITIALIZE LOC. 0
004422  005110                    COM     (R0)
004424  005200                    INC     R0                      ;R0=ODD BYTE
004426  105010                    CLRB    (R0)                    ;TRY TO CLEAR BYTE 1
004430  001401                    BEQ     SOPB1C
004432  104000                    EMT                             ;CLRB DID NOT SET Z-BIT
004434  005300            SOPB1C: DEC     R0                      ;R0=WORD ADDR.
004436  005210                    INC     (R0)                    ;INCREMENT TO TEST WORD
004440  005200                    INC     R0                      ;R0=ODD BYTE
004442  105110                    COMB    (R0)                    ;TRY TO COMPLEMENT BYTE 1
004444  105210                    INCB    (R0)
004446  100002                    BPL     SOPB1D
004450  105210                    INCB    (R0)                    ;TRY TO INCREMENT BYTE 1
004452  001401                    BEQ     TS54
004454                    SOPB1D:
004454  104000                    EMT                             ;TEST CUMMULATIVE RESULT OF ABOVE INST.

                          ;*****************************************************************************
                          ;
                          ;       THIS TEST VERIFIES MODE 2 SINGLE-OPERAND INSTRUCTIONS.  PREVIOUSLY
                          ;TESTED INSTRUCTIONS ARE USED TO SET A POINTER IN R0 TO LOC. 400.
                          ;LOC. 400 IS INITIALIZED TO -1 BEFORE A CLR MODE 2 IS EXECUTED.
                          ;       THEN R0 IS DECREMENTED BY TWO TO AGAIN POINT TO 400 BEFORE EACH
                          ;OF SEVERAL MODE 2 INSTRUCTIONS ARE USED TO VERIFY THE DATA RESULTS OF
                          ;THE TEST.  THIS PROCEDURE ALSO VERIFIES THE PROPER INCREMENTING OF THE
                          ;REGISTER.
                          ;
                          ;*****************************************************************************
                          ;TEST 54         TEST MODE 2 USING SOP INST.
                          ;*****************************************************************************
004456                    TS54:
004456  005000                    CLR     R0                      ;SET R0=400
004460  105100                    COMB    R0
004462  005200                    INC     R0
004464  005010                    CLR     (R0)                    ;CLEAR 400
004466  005110                    COM     (R0)                    ;INITIALIZE: 400=-1
004470  005020                    CLR     (R0)+                   ;TRY CLEARING WITH MODE 2
004472  001401                    BEQ     SOPZA
004474  104000                    EMT                             ;CLR INST DID NOT SET Z-BIT
004476  005300            SOPZA:  DEC     R0                      ;RESET R0
004500  005300                    DEC     R0
004502  005120                    COM     (R0)+                   ;TRY COMPLEMENTING WITH MODE 2
004504  100004                    BPL     SOP2B
004506  005300                    DEC     R0                      ;RESET R0
004510  005300                    DEC     R0
004512  005220                    INC     (R0)+                   ;TRY INCREMENTING WITH MODE 2
004514  001401                    BEQ     TS55
```

```
004516                  SOP2B:
004516  104000                  EMT                     ;CHECK CUMMULATIVE RESULT OF ABOVE INST

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 2 SINGLE OPERAND INSTRUCTIONS WHICH
                        ;ADDRESS EVEN BYTES.  RO IS SET TO 400 AND USED TO INITIALIZE LOCATION
                        ;400 TO -1.  CLRB INSTRUCTION IS THEN EXECUTED ON BYTE 400 WITH
                        ;MODE 2.
                        ;       RO IS THEN DECREMENTED BEFORE EACH OF SEVERAL MODE 2 INSTRUCTIONS
                        ;WHICH ARE USED TO VERIFY THE DATA RESULTS OF THE TEST.  THIS PROCEDURE ALSO
                        ;VERIFIES THE PROPER INCREMENTING OF THE REGISTER.
                        ;
                        ;*******************************************************************************
                        ;TEST 55        TEST MODE 2 EVEN BYTE USING SOP INST.
                        ;*******************************************************************************
004520                  TS55:
004520  005000                  CLR     R0              ;SET R0=400
004522  105100                  COMB    R0
004524  005200                  INC     R0
004526  005010                  CLR     (R0)            ;CLEAR 400
004530  005110                  COM     (R0)            ;INITIALIZE: 400=-1
004532  105020                  CLRB    (R0)+           ;TRY TO CLEAT 400 W/MODE 2
004534  001401                  BEQ     SOPB2A
004536  104000                  EMT                     ;CLR DID NOT SET Z-BIT
004540  005300          SOPB2A: DEC     R0              ;RESULT R0=400
004542  005210                  INC     (R0)            ;INC 400 TO TEST WORD
004544  105110                  COMB    (R0)
004546  105220                  INCB    (R0)+           ;TRY TO INC EVEN BYTE
004550  100003                  BPL     SOPB2B
004552  005300                  DEC     R0              ;RESET R0=400
004554  105220                  INCB    (R0)+           ;TRY INCREMENT OF EVEN BYTE
004556  001401                  BEQ     TS56
004560                  SOPB2B:
004560  104000                  EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INST.

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST FOLLOWS THE SAME PROCEDURE DESCRIBED IN THE PREVIOUS
                        ;TEST.  HERE, THE BYTE INSTRUCTION IS USED TO ADDRESS AN ODD BYTE.
                        ;
                        ;*******************************************************************************
                        ;TEST 56        TEST MODE 2 ODD BYTE USING SOP INST.
                        ;*******************************************************************************
004562                  TS56:
004562  005000                  CLR     R0              ;SET R0=400
004564  105100                  COMB    R0
004566  005200                  INC     R0
004570  005010                  CLR     (R0)            ;CLEAR LOC 400
004572  005110                  COM     (R0)            ;INITIALIZE: 400=-1
004574  005200                  INC     R0              ;R0=ODD BYTE
004576  105020                  CLRB    (R0)+           ;TRY TO CLEAR ODD BYTE
004600  001401                  BEQ     SOPB2C
004602  104000                  EMT                     ;CLRB DID NOT SET Z-BIT
004604  005300          SOPB2C: DEC     R0              ;R0=WORD ADDR.
004606  005300                  DEC     R0
```

```
004610  005220                    INC     (R0)+             ;INCREMENT WORD
004612  005300                    DEC     R0                ;POINT TO ODD BYTE
004614  105110                    COMB    (R0)              ;COMPLEMENT ODD BYTE
004616  105220                    INCB    (R0)+             ;TRY TO INCREMENT ODD BYTE
004620  100003                    BPL     SOPB2D
004622  005300                    DEC     R0                ;RESET R0 TO ODD BYTE
004624  105220                    INCB    (R0)+             ;TRY TO INCREMENT ODD BYTE
004626  001401                    BEQ     TS57
004630                    SOPB2D:
004630  104000                    EMT                       ;TEST CUMMULATIVE RESULT OF ABOVE INST.

                          ;*************************************************************************
                          ;
                          ;       THESE TESTS CHECK THE NEGATE INSTRUCTION IN ALL MODES.  PREVIOUSLY
                          ;TESTED SINGLE-OPERAND INSTRUCTIONS ARE USED TO TEST THE NEGATE INSTRUCTION.
                          ;
                          ;*************************************************************************
                          ;TEST 57        TEST MODE 0 USING NEGATE INSTRUCTION
                          ;*************************************************************************
004632                    TS57:
004632  005000                    CLR     R0                ;SET R0=0
004634  005200                    INC     R0                ;    R0=1
004636  005400                    NEG     R0                ;TRY NEGATE MODE 0:  R0=-1
004640  100003                    BPL     NEG00             ;CC=1001?
004642  001402                    BEQ     NEG00
004644  102401                    BVS     NEG00
004646  103401                    BCS     NEG01
004650                    NEG00:
004650  104000                    EMT                       ;NEGATE DID NOT SET CC'S CORRECTLY

004652  005200            NEG01:  INC     R0                ;TEST DATA RESULT
004654  001401                    BEQ     NEG02
004656  104000                    EMT                       ;DATA RESULT OF NEGATE INCORRECT

004660  105100            NEG02:  COMB    R0                ;R0=377
004662  105400                    NEGB    R0                ;R0=1
004664  100403                    BMI     NEG03             ;CC=0001?
004666  001402                    BEQ     NEG03
004670  102401                    BVS     NEG03
004672  103401                    BCS     NEG04
004674                    NEG03:
004674  104000                    EMT                       ;NEGB DID NOT SET CC'S CORRECTLY
004676  005300            NEG04:  DEC     R0                ;TEST DATA RESULT
004700  001401                    BEQ     TS60
004702  104000                    EMT                       ;DATA RESULT OF NEGB INCORRECT
                          ;*************************************************************************
                          ;TEST 60        TEST MODE 1 USING NEGATE INST.
                          ;*************************************************************************
004704                    TS60:
004704  005000                    CLR     R0                ;POINT TO LOC. 0
004706  005010                    CLR     (R0)              ;CLEAR LOC. 0
004710  005210                    INC     (R0)              ;LOC. 0=1
004712  005410                    NEG     (R0)              ;TRY NEG. LOC. 0=-1
004714  100003                    BPL     NEG10             ;CC=1001
004716  001402                    BEQ     NEG10
004720  102401                    BVS     NEG10
```

```
004722  103401                  BCS     NEG11
004724                  NEG10:
004724  104000                  EMT                     ;NEGATE DID NOT SET CC'S CORRECTLY

004726  005237  000000  NEG11:  INC     @#0             ;TEST DATA RESULT
004732  001401                  BEQ     NEG12
004734  104000                  EMT                     ;DATA RESULT OF NEGATE INCORRECT
004736  105110          NEG12:  COMB    (R0)            ;LOC. 0=377
004740  105410                  NEGB    (R0)            ;TRY NEGB LOC. 0=1
004742  100403                  BMI     NEG13           ;CC=0001?
004744  001402                  BEQ     NEG13
004746  102401                  BVS     NEG13
004750  103401                  BCS     NEG14
004752                  NEG13:
004752  104000                  EMT                     ;NEGB DID NOT SET CC'S CORRECTLY
004754  005337  000000  NEG14:  DEC     @#0             ;TEST DATA RESULT
004760  001401                  BEQ     TS61
004762  104000                  EMT                     ;DATA RESULT OF NEGB INCORRECT
                        ;*****************************************************************************
                        ;TEST 61        TEST MODE 2 USING NEGATE INSTRUCTION
                        ;*****************************************************************************
004764                  TS61:
004764  005000                  CLR     R0              ;POINT TO LOC. 0
004766  005010                  CLR     (R0)            ;CLEAR LOC. 0
004770  005210                  INC     (R0)            ;LOC. 0=1
004772  005420                  NEG     (R0)+           ;TRY NEG.: LOC. 0=-1
004774  100003                  BPL     NEG20           ;CC=1001?
004776  001402                  BEQ     NEG20
005000  102401                  BVS     NEG20
005002  103401                  BCS     NEG21
005004                  NEG20:
005004  104000                  EMT                     ;NEGATE DID NOT SET CC'S CORRECTLY
005006  105300          NEG21:  DECB    R0              ;R0=LOC. 0
005010  105300                  DECB    R0
005012  105420                  NEGB    (R0)+           ;BYTE 0=1   R0=1
005014  105420                  NEGB    (R0)+           ;BYTE 1=1   R0=2
005016  105340                  DECB    -(R0)           ;R0=1  LOC. 0=01
005020  005300                  DEC     R0              ;R0=0
005022  001401                  BEQ     NEG22
005024  104000                  EMT                     ;REGISTER NOT INCREMENTED CORRECTLY
005026  005337  000000  NEG22:  DEC     @#0             ;LOC. 0=0
005032  001401                  BEQ     TS62
005034  104000                  EMT                     ;NEG BYTE INSTRUCTIONS FAILED

                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND INSTRUCTIONS.  IT
                        ;USES LOCATION 0 AS ITS TARGET DATA.  A TABLE LOCATED AT LOC. 400
                        ;THRU 402 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
                        ;INSTRUCTIONS UNDER TEST.
                        ;       R0 IS SET TO 400, THE START OF THE ADDRESS TABLE, AND A CLR
                        ;INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR LOC. 0.  THEN R0
                        ;IS DECREMENTED BY TWO AND TWO OTHER MODE 3 INSTRUCTIONS OPERATE ON
                        ;LOC. 0 TO VERIFY THE DATA RESULTS OF THE TEST.  THE PROPER INCREMENTING
                        ;OF THE REGISTER IS ALSO VERIFIED IN THIS MANNER.
                        ;       IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
```

```
                            ;(LOC. 400-402) HAS THE PROPER VALUES (0).
                            ;
                            ;*******************************************************************************
                            ;TEST 62         TEST MODE 3 USING SOP INST.
                            ;*******************************************************************************
005036                      TS62:
005036  005000                      CLR     R0                      ;SET R0=400
005040  105100                      COMB    R0
005042  005200                      INC     R0
005044  005010                      CLR     (R0)                    ;CLEAR LOC 400
005046  005030                      CLR     @(R0)+                  ;TRY TO CLEAR LOC 0 USING MODE 3 ;R0=402
005050  001401                      BEQ     SOP3A
005052  104000                      EMT                             ;CLR DID NOT SET Z-BIT
005054  005300              SOP3A:  DEC     R0                      ;RESET R0=400
005056  005300                      DEC     R0
005060  005130                      COM     @(R0)+                  ;TRY TO COMPLEMENT LOC 0 OF MODE 3  ;R0=402
005062  100002                      BPL     SOP3B
005064  005230                      INC     @(R0)+                  ;TRY TO INCREMENT LOC 0 W/MODE 3  ;R0=404
005066  001401                      BEQ     TS63
005070                      SOP3B:
005070  104000                      EMT                             ;CUMMULATIVE RESULT OF ABOVE INST FAILED

                            ;*******************************************************************************
                            ;
                            ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
                            ;WHICH ADDRESS EVEN BYTES.  AGAIN, THE TARGET LOCATION 0 IS USED
                            ;AND THE SAME TABLE AT 400 IS EMPLOYED.
                            ;       AFTER POINTING R4 TO THE TABLE (400) AND SETTING LOCATION
                            ;0 TO -1, A CLRB INSTRUCTION IS USED TO CLEAR BYTE 0.
                            ;       SEVERAL OTHER MODE 3 INSTRUCTIONS ARE THEN USED WITH THE TABLE
                            ;TO VERIFY THE DATA RESULTS AND THE PROPER INCREMENTING OF THE REGISTER.
                            ;IF A FAILURE IS DETECTED, BE SURE THAT THE TABLE (LOCATION 400-402) HAS
                            ;THE PROPER VALUES (0).
                            ;
                            ;*******************************************************************************
                            ;TEST 63         TEST MODE 3 EVEN BYTE USING SOP INST.
                            ;*******************************************************************************
005072                      TS63:
005072  005004                      CLR     R4                      ;SET R4=400
005074  105104                      COMB    R4
005076  005204                      INC     R4
005100  005000                      CLR     R0                      ;INITIALIZE LOC. 0=-1
005102  005010                      CLR     (R0)
005104  005110                      COM     (R0)                    ;LOC. 0=-1
005106  105034                      CLRB    @(R4)+                  ;TRY TO CLEAR EVEN BYTE ;LOC. 0=177400  R4=402
005110  001401                      BEQ     SOPB3A
005112  104000                      EMT                             ;CLRB DID NOT SET Z-BIT
005114  005304              SOPB3A: DEC     R4                      ;RESET POINTER  R4=400
005116  005304                      DEC     R4
005120  005234                      INC     @(R4)+                  ;TRY INCREMENTING WORD  LOC.0=177401  R4=402
005122  100006                      BPL     SOPB3B
005124  105434                      NEGB    @(R4)+                  ;TRY TO NEGATE EVEN BYTE ;LOC.0=-1 R4=404
005126  100004                      BPL     SOPB3B
005130  005304                      DEC     R4                      ;R4=402
005132  005304                      DEC     R4
005134  105234                      INCB    @(R4)+                  ;TRY TO INCREMENT EVEN BYTE ;LOC. 0=17400
```

```
005136  001401                  BEQ     TS64
005140                  SOPB3B:
005140  104000                  EMT                     ;CUMMULATIVE RESULT OF ABOVE INST FAILED

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
                        ;WHICH ADDRESS ODD BYTES.  THE TARGET IS BYTE 1.  A TABLE AT
                        ;LOC. 400-406 IS USED.  R0 SERVES AS THE TABLE POINTER.
                        ;       R0 IS INITIALIZED TO 400.  LOC. 0 IS SET TO -1 USING THE
                        ;FIRST TWO TABLE ENTRIES.  A CLRB MODE 3 IS EXECUTED ON BYTE 1 USING
                        ;TABLE ADDRESS AT 404.  R0 IS DECREMENTED TO 402 AND SEVERAL SOP
                        ;MODE 3 INSTRUCTIONS ARE USED TO VERIFY DATA RESULTS AND PROPER
                        ;REGISTER INCREMENTING.
                        ;       THE TABLE (400-406) SHOULD CONTAIN 0,0,1,1 BEFORE AND
                        ;AFTER THE TEST IS RUN.
                        ;
```

```
                            ;*************************************************************************
                            ;TEST 64         TEST MODE 3 ODD BYTE USING SOP INST.
                            ;*************************************************************************
005142                      TS64:
005142  005000                      CLR     R0              ;SET R0=400
005144  105100                      COMB    R0
005146  005200                      INC     R0
005150  005030                      CLR     @(R0)+          ;INITIALIZE
005152  005130                      COM     @(R0)+          ;LOC 0=-1 R0=404
005154  105030                      CLRB    @(R0)+          ;TRY TO CLEAR ODD BYTE LOC. 0=377 R0=406
005156  001401                      BEQ     SOPB3C
005160  104000                      EMT                     ;CLRB DID NOT SET Z-BIT
005162  005300              SOPB3C: DEC     R0              ;RESET R0=402
005164  005300                      DEC     R0
005166  005300                      DEC     R0              ;POINT TO EVEN BYTE ADDR.
005170  005300                      DEC     R0
005172  005230                      INC     @(R0)+          ;INCREMENT WORD LOC. 0=400 R0=404
```

```
005174  105430                   NEGB    @(R0)+          ;TRY TO NEGATE ODD BYTE LOC. 0=177400 R0=406
005176  100002                   BPL     SOPB3D
005200  105230                   INCB    @(R0)+          ;TRY TO INCREMENT ODD BYTE LOC.0=0 R0=410
005202  001401                   BEQ     TS65
005204                   SOPB3D:
005204  104000                   EMT                     ;CUMMULATIVE RESULT OF ABOVE INSTS FAILED
                         ;*******************************************************************************
                         ;TEST 65          TEST MODE 3 USING NEGATE INSTRUCTION
                         ;*******************************************************************************
005206                   TS65:
005206  005000                   CLR     R0              ;R0=400
005210  105100                   COMB    R0
005212  005200                   INC     R0
005214  005010                   CLR     (R0)            ;LOC. 400=0
005216  005004                   CLR     R4              ;R4=0
005220  005014                   CLR     (R4)            ;LOC. 0=0
005222  005214                   INC     (R4)            ;LOC. 0=1
005224  005430                   NEG     @(R0)+          ;TRY NEGATE   LOC. 0=-1  R0=402
005226  100003                   BPL     NEG30           ;CC=1001?
005230  001402                   BEQ     NEG30
005232  102401                   BVS     NEG30
005234  103401                   BCS     NEG31
005236                   NEG30:
005236  104000                   EMT                     ;NEG DID NOT SET CC'S CORRECTLY
005240  005214           NEG31:  INC     (R4)            ;LOC. 0=0
005242  001401                   BEQ     NEG32
005244  104000                   EMT                     ;DATA RESULT OF NEG INCORRECT
005246  105137  000001   NEG32:  COMB    @#1             ;LOC 0=177400
005252  005237  000000           INC     @#0             ;LOC. 0=177401
005256  105430                   NEGB    @(R0)+          ;TRY NEGB LOC. 0=177777  R0=404
005260  100401                   BMI     NEG33
005262  104000                   EMT                     ;NEGB FAILED WITH EVEN BYTE
005264  105430           NEG33:  NEGB    @(R0)+          ;TRY NEGB  LOC.0=777  R0=406
005266  100001                   BPL     NEG34
005270  104000                   EMT                     ;NEGB FAILED WITH ODD BYTE
005272  105137  000001   NEG34:  COMB    @#1             ;LOC. 0=177377
005276  105237  000001           INCB    @#1             ;LOC. 0=177777
005302  005214                   INC     (R4)            ;LOC. 0=0
005304  001401                   BEQ     TS66
005306  104000                   EMT                     ;DATA RESULT OF NEGB'S INCORRECT

                         ;*******************************************************************************
                         ;
                         ;        THIS TEST VERIFIES MODE 4 SINGLE OPERAND INSTRUCTIONS.
                         ;R0 IS SET TO 400. A CLR INSTRUCTION IS EXECUTED IN MODE 4 TO CLEAR
                         ;LOC. 376. R0 IS RESET TO 400 AND A COM INSTRUCTION USING MODE 4
                         ;COMPLEMENTS LOC.376.
                         ;        TWO INC INSTRUCTIONS AND A MODE 4 INSTRUCTION ARE EXECUTED
                         ;TO COMPLETE THE TEST.
                         ;*******************************************************************************
                         ;TEST 66          TEST MODE 4 USING SOP INSTS
                         ;*******************************************************************************
005310                   TS66:
005310  005000                   CLR     R0              ;SET R0=400
005312  105100                   COMB    R0
005314  005200                   INC     R0
```

```
005316  005040                   CLR     -(R0)           ;TRY TO CLEAR USING MODE 4
005320  001401                   BEQ     SOP4A
005322  104000                   EMT                     ;CLR DID NOT SET Z-BIT
005324  005200          SOP4A:   INC     R0              ;RESET R0
005326  005200                   INC     R0
005330  005140                   COM     -(R0)           ;TRY TO COMPLEMENT USING MODE 4
005332  100004                   BPL     SOP4B
005334  005200                   INC     R0              ;MOVE POINTER
005336  005200                   INC     R0
005340  005240                   INC     -(R0)
005342  001401                   BEQ     TS67
005344                  SOP4B:
005344  104000                   EMT                     ;CHECK CUMMULATIVE RESULT OF ABOVE INST.

                        ;*******************************************************************************
                        ;
                        ;        THIS TEST VERIFIES MODE 5 SINGLE OPERAND INSTRUCTIONS.  IT
                        ;USES LOCATION 0 AS ITS TARGET DATA.  A TABLE LOCATED AT LOC. 372
                        ;THRU 374 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
                        ;INSTRUCTIONS UNDER TEST.
                        ;        R0 IS SET TO 376, (THE START OF THE ADDRESS TABLE) +2,
                        ;AND A CLR INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR
                        ;LOC. 0.  THEN R0 IS INCREMENTED BY TWO AND TWO OTHER MODE 3
                        ;INSTRUCTIONS OPERATE ON LOC. 0 TO VERIFY THE DATA RESULTS OF
                        ;THE TEST.  THE PROPER DECREMENTING OF THE REGISTER IS ALSO
                        ;VERIFIED IN THIS MANNER.
                        ;        IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
                        ;(LOC. 372 THRU 374) HAS THE PROPER VALUES (0).
                        ;
                        ;*******************************************************************************
                        ;TEST 67         TEST MODE 5 USING SOP INSTS
                        ;*******************************************************************************
005346                  TS67:
005346  012700  000370           MOV     #370,R0         ;CLEAR LOCATION 370-376
005352  005020                   CLR     (R0)+           ;370
005354  005020                   CLR     (R0)+           ;372
005356  005020                   CLR     (R0)+           ;374
005360  005010                   CLR     (R0)            ;376
005362  005000                   CLR     R0              ;SET R0=376 (LOW BYTE)
005364  005020                   CLR     (R0)+
005366  105400                   NEGB    R0
005370  005050                   CLR     @-(R0)          ;TRY TO CLEAR LOC 0 W/MODE 5
005372  001401                   BEQ     SOP5A
005374  104000                   EMT                     ;CLR DID NOT SET Z-BIT
005376  005200          SOP5A:   INC     R0              ;RESET R0
005400  005200                   INC     R0
005402  005150                   COM     @-(R0)          ;TRY TO COMPLEMENT LOC. 0 W/MODE 5
005404  100002                   BPL     SOP5B
005406  005250                   INC     @-(R0)          ;TRY TO INCREMENT LOC. 0 W/MODE 5
005410  001401                   BEQ     TS70
005412                  SOP5B:
005412  104000                   EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INSTS

                        ;*******************************************************************************
                        ;
                        ;        THIS TEST VERIFIES MODE 6 SINGLE OPERAND INSTRUCTIONS.  IT
```

```
                                 ;USES LOCATION 0 AS ITS TARGET DATA.  R0 IS SET TO 400 USING
                                 ;PREVIOUSLY TESTED INSTRUCTIONS AND A MODE 6 CLR INSTRUCTION IS
                                 ;EXECUTED ON LOC. 0 USING R0 AND A -400 OFFSET.  COM AND INC
                                 ;INSTRUCTIONS ARE THEN USED TO VERIFY THE DATA.
                                 ;
                                 ;*******************************************************************
                                 ;TEST 70         TEST MODE 6 USING SOP INSTS
                                 ;*******************************************************************
005414                           TS70:
005414  005000                          CLR     R0                  ;SET R0=400
005416  105100                          COMB    R0
005420  005200                          INC     R0
005422  005060  177400                  CLR     -400(R0)            ;TRY TO CLEAR LOCATION 0 W/MODE 6
005426  001401                          BEQ     SOP6A
005430  104000                          EMT                         ;CLR DID NOT SET Z-BIT
005432  005160  177400           SOP6A: COM     -400(R0)            ;TRY TO COMPLEMENT LOCATION 0 W/MODE 6
005436  100003                          BPL     SOP6B
005440  005260  177400                  INC     -400(R0)            ;TRY TO INCREMENT LOCATION 0 W/MODE 6
005444  001401                          BEQ     TS71
005446                           SOP6B:
005446  104000                          EMT                         ;TEST CUMMULATIVE RESULT OF ABOVE INSTS

                                 ;*******************************************************************
                                 ;
                                 ;       THIS TEST VERIFIES MODE 7 SINGLE OPERAND INSTRUCTIONS.  IT USES
                                 ;THE POINTER TO LOC. 0 WHICH IS STORED AT LOC. 402.
                                 ;       R0 IS SET TO 400 AND A MODE 7 CLR INSTRUCTION IS
                                 ;EXECUTED WITH A +2 OFFSET TO CLEAR LOC. 0.
                                 ;       SEVERAL OTHER MODE 7 INSTRUCTIONS ARE THEN USED ON THE COMMON
                                 ;LOCATION TO VERIFY THE DATA RESULTS.
                                 ;
                                 ;*******************************************************************
                                 ;TEST 71         TEST MODE 7 USING SOP INST.
                                 ;*******************************************************************
005450                           TS71:
005450  005000                          CLR     R0                  ;SET R0=400
005452  105100                          COMB    R0
005454  005200                          INC     R0
005456  005210                          INC     (R0)                ;R0=1
005460  005070  000002                  CLR     @2(R0)              ;TRY TO CLEAR LOC. 0 W/MODE 7
005464  001401                          BEQ     SOP7A
005466  104000                          EMT                         ;CLR DID NOT SET Z-BIT
005470  005170  000002           SOP7A: COM     @2(R0)              ;TRY TO COMPLEMENT LOC. 0 W/MODE 7
005474  100003                          BPL     SOP7B
005476  005270  000002                  INC     @2(R0)              ;TRY TO INCREMENT LOC. 0 W/MODE 7
005502  001401                          BEQ     TS72
005504                           SOP7B:
005504  104000                          EMT                         ;TEST CUMMULATIVE RESULT OF ABOVE INSTS.

                                 ;*******************************************************************
                                 ;TEST 72         TEST MODE 4 WITH NEGATE INSTRUCTION
                                 ;*******************************************************************
005506                           TS72:
005506  005000                          CLR     R0
005510  005010                          CLR     (R0)
005512  005120                          COM     (R0)+               ;LOC. 0=177777,  R0=2
```

```
005514 005440                 NEG     -(R0)          ;TRY NEGATE, LOC. 0=1
005516 100403                 BMI     NEG40          ;CC=0001?
005520 001402                 BEQ     NEG40
005522 102401                 BVS     NEG40
005524 103401                 BCS     NEG41
005526                NEG40:
005526 104000                 EMT                    ;NEG DID NOT SET CC'S CORRECTLY
005530 005400         NEG41:  NEG     R0             ;TST R0 WITH A NEG.
005532 001401                 BEQ     NEG42
005534 104000                 EMT                    ;R0 NOT DECREMENTED PROPERLY
005536 005310         NEG42:  DEC     (R0)           ;TEST DTA RESULT OF NEG
005540 001401                 BEQ     TS73
005542 104000                 EMT                    ;DATA RESULT OF NEG INCORRECT
                      ;*******************************************************************************
                      ;TEST 73        TEST MODE 5 WITH NEGATE INSTRUCTION
                      ;*******************************************************************************
005544                TS73:
005544 005000                 CLR     R0             ;R0=0
005546 005010                 CLR     (R0)           ;LOC. 0=0
005550 105100                 COMB    R0             ;R0=377
005552 005200                 INC     R0             ;R0=400
005554 005010                 CLR     (R0)           ;SET 400 = 0
005556 005004                 CLR     R4             ;R4=0
005560 005314                 DEC     (R4)           ;LOC. 0=177777
005562 005450                 NEG     @-(R0)         ;TRY NEGATE:  LOC. 0=1
005564 100403                 BMI     NEG50          ;CC=0001?
005566 001402                 BEQ     NEG50
005570 102401                 BVS     NEG50
005572 103401                 BCS     NEG51
005574                NEG50:
005574 104000                 EMT                    ;NEG DID NOT SET CC'S CORRECTLY
005576 005314         NEG51:  DEC     (R4)
005600 001401                 BEQ     NEG52
005602 104000                 EMT                    ;DATA RESULT OF NEG INCORRECT
005604 105100         NEG52:  COMB    R0
005606 005300                 DEC     R0
005610 001401                 BEQ     TS74
005612 104000                 EMT                    ;REGISTER NOT DECREMENTED PROPERLY
                      ;*******************************************************************************
                      ;TEST 74        TEST MODE 6 WITH NEGATE
                      ;*******************************************************************************
005614                TS74:
005614 005000                 CLR     R0             ;R0=0
005616 005004                 CLR     R4             ;R4=0
005620 105100                 COMB    R0             ;R0=377
005622 005014                 CLR     (R4)           ;LOC. 0=0
005624 105024                 CLRB    (R4)+          ;LOC. 0=177777,  R4=1
005626 105114                 COMB    (R4)           ;LOC. 0=177400
005630 005460 177401          NEG     -377(R0)       ;LOC. 0=400
005634 100403                 BMI     NEG60          ;CC=0001
005636 001402                 BEQ     NEG60
005640 102401                 BVS     NEG60
005642 103401                 BCS     NEG61
005644                NEG60:
005644 104000                 EMT                    ;NEG DID NOT SET CC'S CORRECTLY
005646 105314         NEG61:  DECB    (R4)
```

```
005650  001401                  BEQ     TS75
005652  104000                  EMT                     ;DATA RESULT OF NEG INCORRECT
                        ;*************************************************************************
                        ;TEST 75        TEST MODE 7 W/ NEGATE
                        ;*************************************************************************
005654                  TS75:
005654  005000                  CLR     R0              ;R0=0
005656  005010                  CLR     (R0)            ;LOC. 0=0
005660  005110                  COM     (R0)            ;LOC. 0=177777
005662  105100                  COMB    R0              ;R0=377
005664  105470  000005          NEGB    @5(R0)          ;R0+5=404,  404=1,  LOC. 0=777
005670  100403                  BMI     NEG70           ;CC=0001?
005672  001402                  BEQ     NEG70
005674  102401                  BVS     NEG70
005676  103401                  BCS     NEG71
005700                  NEG70:
005700  104000                  EMT                     ;NEG DID NOT SET CC'S CORRECTLY
005702  105100          NEG71:  COMB    R0              ;R0=0
005704  105120                  COMB    (R0)+           ;LOC. 0=400,  R0=1
005706  105310                  DECB    (R0)            ;LOC. 0=0
005710  005467  172064          NEG     0               ;USE NEG MODE 67 TO TST FOR ZERO
005714  001401                  BEQ     TS76
005716  104000                  EMT                     ;DATA RESULT OF NEG WAS INCORRECT

                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES PROGRAM COUNTER ADDRESSING WITH SOP
                        ;INSTRUCTIONS.  CLR MODE 77 IS USED TO CLEAR THE LOCATION FOLLOWING THE
                        ;INSTRUCTION (SOPX).  THEN SINGLE OPERAND INSTRUCTIONS WITH MODES 37, 67, AND
                        ;77, USING INDIRECT POINTER SOPXAD ARE USED TO VERIFY THE DATA RESULTS
                        ;OF THESE INSTRUCTIONS.
                        ;
                        ;*************************************************************************
                        ;TEST 76        TEST SOP INSTRUCTIONS MODES 2,3,6,7 WITH REGISTER 7
                        ;*************************************************************************
005720                  TS76:
005720  005027                  CLR     (R7)+           ;CLEAR NEXT LOCATION: (SOPX)
005722  177777          SOPX:   -1                      ;USE MODE 27
005724  001401                  BEQ     SOPA
005726  104000                  EMT                     ;CLR DID NOT SET Z-BIT
005730  005237  005722  SOPA:   INC     @#SOPX          ;INC SOPX W/MODE 37
005734  005467  177762          NEG     SOPX            ;NEGATE SOPX W/MODE 67
005740  100003                  BPL     SOPB
005742  005277  000004          INC     @SOPXAD         ;INC SOPX W/MODE 77
005746  001402                  BEQ     TS77
005750                  SOPB:
005750  104000                  EMT                     ;INC DID NOT SET Z-BIT
005752  005722          SOPXAD: SOPX                    ;INDIRECT ADDRESS OF SOPX

                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING INSTRUCTIONS
                        ;USING MODE 0.  R0 IS SET TO ZERO AND THE CONDITION CODES ARE SET
                        ;TO THE COMPLEMENT OF THAT EXPECTED BY THE INSTRUCTION.  A TST INSTRUCTION
                        ;IS EXECUTED AND CONDITIONAL BRANCHES ARE USED TO TEST THE CONDITION
                        ;CODES.
```

```
                        ;
                        ;*******************************************************************************
                        ;TEST 77         TEST MODE 0 SOP NON-MODIFYING
                        ;*******************************************************************************
005754                  TS77:
005754  005000                  CLR     R0                      ;INITIALIZE R0=0
005756  000277                  SCC                             ;SET CC=1011
005760  000244                  CLZ
005762  005700                  TST     R0                      ;TRY TST W/ MODE 0
005764  102403                  BVS     SNM0A                   ;CHECK T/ T CC=0100
005766  100402                  BMI     SNM0A
005770  103401                  BCS     SNM0A
005772  001401                  BEQ     TS100
005774                  SNM0A:
005774  104000                  EMT                             ;CONDITION CODES NOT SET PROPERLY

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING BYTE INSTRUCTIONS WITH MODE 0.
                        ;R0 IS SET TO 377 AND COMPLEMENT OF THE EXPECTED CONDITION CODES
                        ;IS LOADED IN PSW.  A TSTB INSTRUCTION IS EXECUTED AND THE RESULTS
                        ;ARE CHECKED WITH SEVERAL CONDITIONAL BRANCH INSTRUCTIONS.
                        ;       THIS VERIFIES THAT THE PROPER BYTE WAS TESTED.
                        ;
                        ;*******************************************************************************
                        ;TEST 100        TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING
                        ;*******************************************************************************
005776                  TS100:
005776  005000                  CLR     R0                      ;INITIALIZE
006000  105100                  COMB    R0                      ;R0=377
006002  000277                  SCC                             ;SET CC=0111
006004  000250                  CLN
006006  105700                  TSTB    R0                      ;TRY TST EVEN BYTE
006010  102402                  BVS     SNMB0A                  ;CHECK CC=1000
006012  101401                  BLOS    SNMB0A
006014  100401                  BMI     TS101
006016                  SNMB0A:
006016  104000                  EMT                             ;CONDITION CODES NOT SET PROPERLY

                        ;*******************************************************************************
                        ;
                        ;       THIS TEST VERIFIES SINGLE OPERAND INSTRUCTIONS WITH MODE 1.
                        ;R0 IS USED TO POINT TO AND CLEAR LOC. 0.  THE COMPLEMENT OF THE
                        ;EXPECTED CONDITION CODES ARE LOADED IN THE PSW.  A TST INSTRUCTION
                        ;IS THEN EXECUTED ON LOC. 0 USING R0 AND CONDITIONAL BRANCHES TEST
                        ;THE RESULTS.
                        ;
                        ;*******************************************************************************
                        ;TEST 101        TEST MODE 1 SOP NON-MODIFYING
                        ;*******************************************************************************
006020                  TS101:
006020  005000                  CLR     R0                      ;POINT TO LOC 0
006022  005010                  CLR     (R0)                    ;CLEAR LOC 0
006024  000277                  SCC                             ;INITIALIZE
006026  000244                  CLZ                             ;CC=1011
006030  005710                  TST     (R0)                    ;TRY TST W/ MODE 1
```

```
006032  102403          BVS     SNM1A           ;CHECK CC=0100
006034  103402          BCS     SNM1A
006036  100401          BMI     SNM1A
006040  001401          BEQ     TS102
006042          SNM1A:
006042  104000          EMT                     ;CC'S NOT SET PROPERLY

                ;*******************************************************************************
                ;
                ;       THIS TEST SETS LOCATION 0 TO 377 AND THEN USES R0 TO TEST
                ;THE EVEN BYTE AND THE ODD BYTE USING SOP BYTE INSTRUCTIONS WITH MODE 1.
                ;AGAIN, CONDITIONAL BRANCHES ARE USED TO VERIFY THE SETTING OF THE
                ;PROPER CONDITION CODE BITS.
                ;
                ;*******************************************************************************
                ;TEST 102       TEST MODE 1 BYTE INST. NON-MODIFYING
                ;*******************************************************************************
006044          TS102:
006044  005000          CLR     R0              ;POINT TO LOC 0
006046  005010          CLR     (R0)            ;CLEAR LOC 0
006050  105110          COMB    (R0)            ;COMPLEMENT BYTE 0
006052  000277          SCC                     ;SET CC=0111
006054  000250          CLN
006056  105710          TSTB    (R0)            ;TRY TST ON EVEN BYTE
006060  102402          BVS     SNMB1A
006062  101401          BLOS    SNMB1A
006064  100401          BMI     SNMB1B
006066          SNMB1A:
006066  104000          EMT                     ;CC'S NOT CORRECT
006070  005000  SNMB1B: CLR     R0
006072  005200          INC     R0
006074  000277          SCC                     ;SET CC=1011
006076  000244          CLZ
006100  105710          TSTB    (R0)            ;TRY TO TST AN ODD BYTE
006102  102403          BVS     SNMB1C          ;CHECK CC=0100
006104  103402          BCS     SNMB1C
006106  100401          BMI     SNMB1C
006110  001401          BEQ     TS103
006112          SNMB1C:
006112  104000          EMT                     ;CC'S NOT CORRECT

                ;*******************************************************************************
                ;
                ;       THIS TEST VERIFIES THE SINGLE-OPERAND NON-MODIFYING INSTRUCTIONS
                ;USING MODE 2.  IT USES THE IDENTICAL PROCEDURE EMPLOYED IN THE
                ;MODE 1 TESTS.  ADDITIONALLY, THE REGISTER IS CHECKED TO ASSURE THAT
                ;IT IS INCREMENTED PROPERLY.
                ;
                ;*******************************************************************************
                ;TEST 103       TEST MODE 2 WITH SOP NON-MODIFYING
                ;*******************************************************************************
006114          TS103:
006114  005000          CLR     R0              ;INITIALIZE R0=0
006116  005010          CLR     (R0)            ;CLEAR LOC 0
006120  000277          SCC                     ;SET CC=1011
006122  000244          CLZ
```

```
006124  005720                    TST     (R0)+                   ;TRY TST W/ MODE 2
006126  102403                    BVS     SNM2A                   ;CHECK CC=0100
006130  103402                    BCS     SNM2A
006132  100401                    BMI     SNM2A
006134  001401                    BEQ     SNM2B
006136                    SNM2A:
006136  104000                    EMT                             ;CC'S NOT CORRECT
006140  005300            SNM2B:  DEC     R0                      ;RESET R0
006142  005300                    DEC     R0
006144  001401                    BEQ     TS104
006146  104000                    EMT                             ;MODE 2 DID NOT INC REQ CORRECTLY

                          ;*******************************************************************************
                          ;
                          ;       THIS TEST VERIFIES MODE 2 SINGLE OPERAND NON-MODIFYING BYTE
                          ;INSTRUCTIONS IT USES R0 TO POINT TO LOC. 0.  WITH LOCATION 0
                          ;SET TO 377, THE EVEN AND ODD BYTE IS TESTED WITH TSTB INSTRUCTIONS
                          ;TO VERIFY THE CORRECT CC ARE SET.  THE REGISTER IS CHECKED FOR
                          ;PROPER INCREMENTING.
                          ;
                          ;*******************************************************************************
                          ;TEST 104       TEST MODE 2 - BYTE W/ SOP NON-MODIFYING
                          ;*******************************************************************************
006150                    TS104:
006150  005000                    CLR     R0                      ;CLEAR R0
006152  005010                    CLR     (R0)                    ;CLEAR LOC 0
006154  105110                    COMB    (R0)                    ;SET LOC 0=377
006156  000277                    SCC                             ;SET CC=0111
006160  000250                    CLN
006162  105720                    TSTB    (R0)+                   ;TRY TST OF EVEN BYTE
006164  102402                    BVS     SNMB2A
006166  101401                    BLOS    SNMB2A
006170  100401                    BMI     SNMB2B
006172                    SNMB2A:
006172  104000                    EMT                             ;CC'S NOT SET CORRECTLY
006174  005300            SNMB2B: DEC     R0                      ;DECREMENT R0
006176  001401                    BEQ     SNMB2C
006200  104000                    EMT                             ;MODE 2 DID NOT INC REG CORRECTLY
006202  005200            SNMB2C: INC     R0                      ;POINT TO ODD BYTE
006204  000277                    SCC                             ;SET CC=1011
006206  000244                    CLZ
006210  105720                    TSTB    (R0)+                   ;TRY TST OF ODD BYTE
006212  102403                    BVS     SNMB2D                  ;CHECK CC'S=0100
006214  103402                    BCS     SNMB2D
006216  100401                    BMI     SNMB2D
006220  001401                    BEQ     SNMB2E
006222                    SNMB2D:
006222  104000                    EMT                             ;CC'S NOT CORRECT
006224  005300            SNMB2E: DEC     R0
006226  005300                    DEC     R0
006230  001401                    BEQ     TS105
006232  104000                    EMT                             ;R0 DID NOT INCREMENT PROPERLY

                          ;*******************************************************************************
                          ;
                          ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND NON-MODIFYING INSTRUCTIONS.
```

```
                          ;A POINTER IN A TABLE AT LOC. 376 IS USED TO TEST LOCATION 0.
                          ;THE CC'S AND THE REGISTER ARE CHECKED FOLLOWING THE
                          ;TST MODE 3 INSTRUCTION.
                          ;
                          ;*************************************************************************
                          ;TEST 105        TEST MODE 3 W/ SOP NON-MODIFYING INSTS
                          ;*************************************************************************
006234                    TS105:
006234  005000                     CLR     R0              ;R0=0
006236  005010                     CLR     (R0)            ;CLEAR LOC 0
006240  105100                     COMB    R0              ;R0=376
006242  005300                     DEC     R0
006244  000277                     SCC                     ;SET CC=1011
006246  000244                     CLZ
006250  005730                     TST     @(R0)+          ;TRY TST W/ MODE 3
006252  102403                     BVS     SNM3A           ;CHECK CC=0100
006254  103402                     BCS     SNM3A
006256  100401                     BMI     SNM3A
006260  001401                     BEQ     SNM3B
006262                    SNM3A:
006262  104000                     EMT                     ;CC'S NOT CORRECT
006264  005300            SNM3B:   DEC     R0              ;R0=377
006266  105100                     COMB    R0              ;R0=0
006270  001401                     BEQ     TS106
006272  104000                     EMT                     ;MODE 3 DID NOT INC REG CORRECTLY

                          ;*************************************************************************
                          ;
                          ;        THIS TEST VERIFIES SOP NON-MODIFYING BYTE INSTRUCTIONS MODE 3
                          ;LOC. 0 IS SET TO 377.  TABLE AT LOC. 402-404 IS USED TO TEST
                          ;BYTE 0 AND BYTE 1.  THE REGISTER IS CHECKED FOR PROPER INCREMENTING AND
                          ;THE CC'S ARE VERIFIED.
                          ;        THE TABLE AT LOC. 402-404 SHOULD CONTAIN 0 AND 1 BEFORE AND
                          ;AFTER THE TEST IS RUN.
                          ;
                          ;*************************************************************************
                          ;TEST 106        TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.
                          ;*************************************************************************
006274                    TS106:
006274  005000                     CLR     R0              ;R0=0
006276  005010                     CLR     (R0)            ;CLEAR LOC 0
006300  105110                     COMB    (R0)            ;LOC. 0 =377
006302  105100                     COMB    R0
006304  005200                     INC     R0
006306  005720                     TST     (R0)+           ;R0=402
006310  000277                     SCC                     ;CC=0111
006312  000250                     CLN
006314  105730                     TSTB    @(R0)+          ;TRY TST OF EVEN BYTE
006316  102402                     BVS     SNMB3A          ;CHECK CC=1000
006320  101401                     BLOS    SNMB3A
006322  100401                     BMI     SNMB3B
006324                    SNMB3A:
006324  104000                     EMT                     ;CC'S NOT CORRECT
006326  000277            SNMB3B:  SCC                     ;SET CC=1011
006330  000244                     CLZ
006332  105730                     TSTB    @(R0)+          ;TRY TST OF ODD BYTE
```

```
006334  102403                BVS     SNMB3C          ;CHECK CC=0100
006336  103402                BCS     SNMB3C
006340  100401                BMI     SNMB3C
006342  001401                BEQ     SNMB3D
006344                SNMB3C:
006344  104000                EMT                     ;CC'S NOT CORRECT
006346  005720        SNMB3D: TST     (R0)+           ;R0=410
006350  005710                TST     (R0)
006352  100401                BMI     TS107
006354  104000                EMT                     ;TSTB DID NOT INCREMENT R0 CORRETCLY
                      ;*************************************************************************
                      ;
                      ;       THIS TEST VERIFIES MODE 4 SOP NON-MODIFYING INSTRUCTIONS.
                      ;LOC. 0 IS SET TO -1 AND THE CC'S ARE SET TO THE COMPLEMENT OF THE
                      ;EXPECTED RESULTS.  R0 AND SET TO 2 AND A TST MODE 4 IS EXECUTED.
                      ;THE CC'S ARE CHECKED WITH CONDITIONAL BRANCH INSTRUCTIONS AND THE REGISTER
                      ;IS CHECKED FOR PROPER DECREMENTING.
                      ;
                      ;*************************************************************************
                      ;TEST 107       TEST MODE 4 W/ SOP NON-MODIFYING INSTS
                      ;*************************************************************************
006356                TS107:
006356  005000                CLR     R0              ;R0=0
006360  005010                CLR     (R0)            ;LOC 0=0
006362  005120                COM     (R0)+           ;LOC 0=-1
006364  000277                SCC                     ;SET CC=1011
006366  000244                CLZ
006370  005740                TST     -(R0)           ;TRY TST W/ MODE 4
006372  102402                BVS     SNM4A           ;CHECK CC=0100
006374  101401                BLOS    SNM4A
006376  100401                BMI     SNM4B
006400                SNM4A:
006400  104000                EMT                     ;CC'S NOT CORRECT
006402  005700        SNM4B:  TST     R0
006404  001401                BEQ     TS110
006406  104000                EMT                     ;TST MODE 4 DID NOT DEC R0 CORRECTLY
                      ;*************************************************************************
                      ;
                      ;       THIS TEST VERIFIES MODE 5 SOP NON-MODIFYING INSTRUCTIONS.
                      ;IT USES A POINTER AT LOC. 376 TO TEST LOC. 0.  R0 IS SET
                      ;TO 400, A TST MODE 5 INSTRUCTION IS EXECUTED AND THE CC'S CHECKED.
                      ;R0 IS CHECKED TO INSURE PROPER DECREMENTING.
                      ;
                      ;*************************************************************************
                      ;TEST 110       TEST MODE 5 W/ SOP NON-MODIFYING INSTS
                      ;*************************************************************************
006410                TS110:
006410  005000                CLR     R0              ;R0=0
006412  005010                CLR     (R0)            ;LOC 0=0
006414  005110                COM     (R0)            ;LOC 0=-1
006416  105100                COMB    R0              ;R0=377
006420  005200                INC     R0              ;R0=400
006422  000277                SCC                     ;SET CC=0111
006424  000250                CLN
006426  005750                TST     @-(R0)          ;TRY TST W/ MODE 5
```

```
006430  102402                         BVS     SNM5A           ;CHECK CC=1000
006432  101401                         BLOS    SNM5A
006434  100401                         BMI     SNM5B
006436                         SNM5A:
006436  104000                         EMT                     ;CC'S NOT SET PROPERLY
006440  005200                 SNM5B:  INC     R0              ;R0=377
006442  105100                         COMB    R0              ;R0=0
006444  001401                         BEQ     TS111
006446  104000                         EMT                     ;MODE 5 DID NOT DEC R0 CORRECTLY

                               ;*****************************************************************************
                               ;
                               ;       THIS TEST VERIFIES MODE 6 SOP NON-MODIFYING INSTRUCTIONS.
                               ;R0 IS SET TO 377 AND A MODE 6 TST INSTRUCTION IS EXECUTED
                               ;USING R0 AND AN OFFSET OF -377.  THE CC'S ARE CHECKED AS WELL
                               ;AS R0 TO INSURE IT WAS NOT ALTERED.
                               ;
                               ;*****************************************************************************
                               ;TEST 111       TEST MODE 6 W/ SOP NON-MODIFYING INSTS
                               ;*****************************************************************************
006450                         TS111:
006450  005000                         CLR     R0              ;R0=0
006452  005010                         CLR     (R0)            ;LOC 0=0
006454  005110                         COM     (R0)            ;LOC 0=-1
006456  105100                         COMB    R0              ;R0=377
006460  000277                         SCC                     ;SET CC=0111
006462  000250                         CLN
006464  005760  177401                 TST     -377(R0)        ;TRY TST W/ MODE 6
006470  102402                         BVS     SNM6A           ;CHECK CC=1000
006472  101401                         BLOS    SNM6A
006474  100401                         BMI     SNM6B
006476                         SNM6A:
006476  104000                         EMT                     ;CC'S INCORRECT
006500  105100                 SNM6B:  COMB    R0              ;R0=0
006502  001401                         BEQ     TS112
006504  104000                         EMT                     ;TST MODE 6 INCORRECTLY CHANGED R0
```

```

                                   ;*************************************************************************
                                   ;
                                   ;       THIS TEST VERIFIES MODE 7 SOP NON-MODIFYING INSTRUCTIONS.
                                   ;IT USES A POINTER TO LOC. 0 STORED AT LOC. 400 TO TST LOC. 0.
                                   ;R0 IS SET TO 377 AND LOC. 0 IS TESTED THRU THE POINTER AT 400 USING
                                   ;R0 AND AN OFFSET OF 1.
                                   ;
                                   ;*************************************************************************
                                   ;TEST 112        TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
                                   ;*************************************************************************
 006506                            TS112:
 006506  005000                            CLR     R0              ;R0=0
 006510  005010                            CLR     (R0)            ;LOC 0=0
 006512  005110                            COM     (R0)            ;LOC 0=-1
 006514  105100                            COMB    R0              ;R0=377
 006516  000277                            SCC                     ;CC=0111
 006520  000250                            CLN
 006522  005770  000001                    TST     @1(R0)          ;TRY TST W/ MODE 7
 006526  102402                            BVS     SNM7A           ;CHECK CC=1000
 006530  101401                            BLOS    SNM7A
 006532  100401                            BMI     SNM7B
 006534                            SNM7A:
 006534  104000                            EMT                     ;CC'S NOT CORRECT
 006536  105100                    SNM7B:  COMB    R0              ;R0=0
 006540  001401                            BEQ     TS113
 006542  104000                            EMT                     ;TST MODE 7 INCORRECTLY CHANGED R0

                                   ;*************************************************************************
                                   ;
                                   ;       THIS TEST VERIFIES MODE 0 DOUBLE OPERAND INSTRUCTIONS.  IT SETS
                                   ;DATA IN R0 AND R4 AND USES THE ADD INSTRUCTION TO TEST THE DOP
                                   ;MICROCODE.
                                   ;
                                   ;*************************************************************************
                                   ;TEST 113        TEST MODE 0 DOUBLE-OPERAND (DOP) INSTS.
                                   ;*************************************************************************
 006544                            TS113:
 006544  005000                            CLR     R0              ;R0=0
 006546  005100                            COM     R0              ;R0=-1
 006550  005004                            CLR     R4              ;R4=0
 006552  060004                            ADD     R0,R4           ;TRY ADD: R4=-1
 006554  005204                            INC     R4              ;R4=0
 006556  001401                            BEQ     TS114
 006560  104000                            EMT                     ;ADD INST. FAILED W/ MODE 0

                                   ;*************************************************************************
                                   ;
                                   ;       THIS TEST VERIFIES THE MOVE INSTRUCTION WITH MODE 0 TO MODE 0.
                                   ;
                                   ;*************************************************************************
                                   ;TEST 114        MOV MODE 0 TO MODE 0
                                   ;*************************************************************************
 006562                            TS114:
 006562  005000                            CLR     R0              ;R0=0
 006564  005004                            CLR     R4              ;R4=0
```

```
006566  005100                  COM     R0              ;R0=-1
006570  010004                  MOV     R0,R4           ;TRY MOVE -1 TO R4
006572  005204                  INC     R4              ;INC R4
006574  001401                  BEQ     TS115
006576  104000                  EMT                     ;MOVE FAILED MODE 0 TO MODE 0

                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THE SUBTRACT INSTRUCTION WITH MODE 0,0.
                        ;
                        ;*****************************************************************************
                        ;TEST 115       TEST SUB MODE 0,0
                        ;*****************************************************************************
006600                  TS115:
006600  005000                  CLR     R0              ;R0=0
006602  005004                  CLR     R4              ;R4=0
006604  005204                  INC     R4              ;R4=1
006606  160400                  SUB     R4,R0           ;TRY SUB 0,0   R0=-1
006610  100003                  BPL     SUB0            ;CC=1001
006612  001402                  BEQ     SUB0
006614  102401                  BVS     SUB0
006616  103401                  BCS     SUB0A
006620                  SUB0:
006620  104000                  EMT                     ;CONDITION CODE FAILED ON SUB
006622  005200          SUB0A:  INC     R0
006624  001401                  BEQ     TS116
006626  104000                  EMT                     ;DATA RESULT OF SUB FAILED

                        ;*****************************************************************************
                        ;
                        ;       THIS TEST QUICKLY VERIFIES THE REMAINING DOP MODIFYING INSTRUCTIONS
                        ;WITH MODE 0,0 TO PROVIDE A BASELINE FOR SUBSEQUENT TESTS.
                        ;SINGLE OPERAND INSTRUCTIONS ARE USED TO SET UP DATA IN R0 AND R4
                        ;BEFORE EACH OF THE SEVERAL DOP MODIFYING INSTRUCTIONS ARE USED AND
                        ;VERIFIED.
                        ;
                        ;*****************************************************************************
                        ;TEST 116       TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
                        ;*****************************************************************************
006630                  TS116:
006630  005000                  CLR     R0              ;R0=0
006632  010004                  MOV     R0,R4           ;TRY MOVE MODE 0,0
006634  001401                  BEQ     DOP0A
006636  104000                  EMT                     ;Z-BIT NOT SET
006640  005200          DOP0A:  INC     R0              ;R0=1
006642  005100                  COM     R0              ;R0=177776
006644  005104                  COM     R4              ;R4=177777
006646  040004                  BIC     R0,R4           ;TRY BIC: R4=1
006650  005304                  DEC     R4              ;R4=0
006652  001401                  BEQ     DOP0B
006654  104000                  EMT                     ;BIC CLEAR RESULT INCORRECT
006656  050004          DOP0B:  BIS     R0,R4           ;TRY BIS: R4=177777
006660  005204                  INC     R4
006662  005204                  INC     R4              ;R4=0
006664  001401                  BEQ     DOP0C
006666  104000                  EMT                     ;RESULT OF BIS INCORRECT
```

```
006670  005000                  DOPOC:  CLR     R0              ;R0=0
006672  105100                          COMB    R0              ;R0=377
006674  005004                          CLR     R4              ;R4=0
006676  005104                          COM     R4              ;R4=177777
006700  040004                          BIC     R0,R4           ;R4=177400
006702  060004                          ADD     R0,R4           ;TRY ADD: R4=177777
006704  005204                          INC     R4              ;R4=0
006706  001401                          BEQ     DOPOD
006710  104000                          EMT                     ;RESULT OF ADD INCORRECT
006712  160004                  DOPOD:  SUB     R0,R4           ;177401=R4
006714  105404                          NEGB    R4              ;R4=177777
006716  005204                          INC     R4              ;RD=0
006720  001401                          BEQ     TS117
006722  104000                          EMT                     ;RESULT OF SUB INCORRECT

                                ;***********************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND INSTRUCTIONS.  IT SETS
                                ;DATA IN R0 AND LOCATION 0 AND OPERATES UPON IT USING DOP INSTRUCTIONS.
                                ;
                                ;***********************************************************************
                                ;TEST 117       TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS
                                ;***********************************************************************
006724                          TS117:
006724  005000                          CLR     R0              ;R0=0
006726  005010                          CLR     (R0)            ;LOC. 0=0
006730  105110                          COMB    (R0)            ;LOC. 0=377
006732  005220                          INC     (R0)+           ;LOC. 0=400    R0=2
006734  005400                          NEG     R0              ;R0=-2
006736  060037  000000                  ADD     R0,@#0          ;TRY ADD 0,3;  LOC. 0=376
006742  100403                          BMI     DOP03A          ;CC=0001?
006744  001402                          BEQ     DOP03A
006746  102401                          BVS     DOP03A
006750  103401                          BCS     DOP03B
006752                          DOP03A:
006752  104000                          EMT                     ;CC'S NOT SET CORRECTLY
006754  105137  000000          DOP03B: COMB    @#0             ;LOC. 0=1
006760  005337  000000                  DEC     @#0             ;LOC. 0=0
006764  001401                          BEQ     TS120
006766  104000                          EMT                     ;DATA RESULT INCORRECT
                                ;***********************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 0,0 DOP NON-MODIFYING INSTRUCTIONS.
                                ;R0 AND R4 ARE PRESET TO 0 AND 1 RESPECTIVELY.  COMPARE INSTRUCTIONS ARE
                                ;THEN EXECUTED AND CHECKED.  FIRST R4 IS COMPARED TO R0 THEN R0 TO R4.
                                ;
                                ;***********************************************************************
                                ;TEST 120       TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0
                                ;***********************************************************************
006770                          TS120:
006770  005000                          CLR     R0              ;R0=0
006772  005004                          CLR     R4              ;R4=0
006774  005204                          INC     R4              ;R4=1
006776  020400                          CMP     R4,R0           ;TRY COMPARE R4 TO R0
007000  003001                          BGT     DNM1
007002  104000                          EMT                     ;CC'S NOT CORRECT FOR CMP
```

```
007004  020004          DNM1:   CMP     R0,R4           ;TRY COMPARE R0 TO R4
007006  002401                  BLT     DNM2
007010  104000                  EMT                     ;CC'S NOT CORRECT FOR CMP
007012  005200          DNM2:   INC     R0              ;R0=1
007014  020400                  CMP     R4,R0           ;TRY COMPARE R4=1 TO R0=1
007016  001401                  BEQ     DNM3
007020  104000                  EMT                     ;CC'S NOT CORRECT (Z=1) FOR CMP
007022  005000          DNM3:   CLR     R0              ;R0=0
007024  005100                  COM     R0              ;R0=177777
007026  005004                  CLR     R4              ;R4=0
007030  030004                  BIT     R0,R4           ;TRY BIT R0 TO R4
007032  001401                  BEQ     DNM4
007034  104000                  EMT                     ;CC'S NOT CORRECT FOR BIT
007036  005304          DNM4:   DEC     R4              ;R4=177777
007040  030004                  BIT     R0,R4           ;TRY BIT AGAIN
007042  100401                  BMI     TS121
007044  104000                  EMT                     ;CC'S NOT CORRECT FOR BIT
                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND NON-MODIFYING INSTRUCTIONS.
                        ;IT SETS DATA IN R0 AND LOCATION 0 AND COMPARES THEM USING DOPNM INSTRUCTIONS.
                        ;
                        ;*************************************************************************
                        ;TEST 121       TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
                        ;*************************************************************************
007046                  TS121:
007046  005000                  CLR     R0              ;R0=0
007050  005010                  CLR     (R0)            ;LOC. 0=0
007052  005110                  COM     (R0)            ;LOC. 0=177777
007054  005200                  INC     R0              ;R0=1
007056  020037  000000          CMP     R0,@#0          ;TRY CMP MODE 0,3
007062  100403                  BMI     DNM03A          ;CC=0001
007064  001402                  BEQ     DNM03A
007066  102401                  BVS     DNM03A
007070  103401                  BCS     DNM03B
007072                  DNM03A:
007072  104000                  EMT                     ;CC'S NOT SET CORRECTLY
007074  005300          DNM03B: DEC     R0
007076  001002                  BNE     DNM03C
007100  005210                  INC     (R0)
007102  001401                  BEQ     TS122
007104                  DNM03C:
007104  104000                  EMT                     ;DATA INCORRECTLY MODIFIED BY CMP
                        ;*************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS.  R0 IS SET TO -1
                        ;AND LOC 0 TO 1.  R4 IS THEN CLEARED AND USED TO POINT TO LOC 0.
                        ;IN THE ADD MODE 1 INSTRUCTION, LOC 0 IS ADDED TO R0 AND THE
                        ;RESULTS VERIFIED.
                        ;
                        ;*************************************************************************
                        ;TEST 122       TEST MODE 1 W/ DOP INST.
                        ;*************************************************************************
007106                  TS122:
007106  005000                  CLR     R0              ;R0=0
007110  005100                  COM     R0              ;R0=177777
```

```
007112  005004          CLR     R4              ;R4=0
007114  005014          CLR     (R4)            ;LOC 0=0
007116  005214          INC     (R4)            ;LOC 0=1
007120  061400          ADD     (R4),R0         ;TRY ADD SOURCE MODE 1
007122  001401          BEQ     TS123
007124  104000          EMT                     ;RESULT OF ADD INCORRECT

                ;*******************************************************************************
                ;
                ;       THIS TEST VERIFIES MODE 1 DOP BYTE INSTRUCTIONS WHICH ADDRESS
                ;EVEN BYTES.  LOC. 0 IS SET TO -1 AND R4 IS CLEARED.  THEN R4 IS
                ;SET TO -1 USING A BISB THRU R0 WITH MODE 1.
                ;
                ;*******************************************************************************
                ;TEST 123       TEST MODE 1 - EVEN BYTE W/ DOP INSTS.
                ;*******************************************************************************
007126          TS123:
007126  005000          CLR     R0              ;R0=0
007130  005010          CLR     (R0)            ;LOC. 0=0
007132  005110          COM     (R0)            ;LOC. 0=177777
007134  005004          CLR     R4              ;R4=0
007136  151004          BISB    (R0),R4         ;TRY MODE 1- EVEN BYTE W/ DOP
007140  105104          COMB    R4              ;R4=0
007142  001401          BEQ     TS124
007144  104000          EMT                     ;RESULT OF BISB IS INCORRECT

                ;*******************************************************************************
                ;
                ;       THIS TEST VERIFIES MODE 1 DOP NON-MODIFYING INSTRUCTIONS
                ;WHICH ADDRESS EVEN BYTES.  LOC. 0 IS SET TO -1 AND R0 IS CLEARED
                ;AND USED AS THE ADDRESSING REGISTER.  R4 IS SET TO 377 AND A
                ;MODE 1,0 CMPB INSTRUCTION IS USED THE RESULTS VERIFIED.
                ;
                ;*******************************************************************************
                ;TEST 124       TEST MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING INST.
                ;*******************************************************************************
007146          TS124:
007146  005000          CLR     R0              ;R0=0
007150  005010          CLR     (R0)            ;LOC 0=0
007152  005110          COM     (R0)            ;LOC 0=177777
007154  005004          CLR     R4              ;R4=0
007156  105104          COMB    R4              ;R4=377
007160  121004          CMPB    (R0),R4         ;TRY MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING
007162  001401          BEQ     TS125
007164  104000          EMT                     ;RESULT OF CMPB INCORRECT

                ;*******************************************************************************
                ;
                ;       THIS TEST VERIFIES MODE 1,0 MOVB INSTRUCTIONS
                ;WHICH ADDRESS EVEN BYTES.  LOC. 0 IS SET TO 177400, R0 IS CLEARED AND
                ;R4 IS SET TO -1.  MOVB ARE USED TO MOVE BYTE 0 TO R4.  THIS
                ;VERIFIES THAT THE PROPER BYTE WAS SELECTED AND THAT THE SIGN-X-TEND
                ;FUNCTION WITH MODE 0.
                ;       THEN LOC. 0 IS COMPLEMENTED AND THE SAME PROCEDURE EXERCISES
                ;THE LOGIC FOR COMPLEMENTARY DATA.
                ;       THIS TEST EXERCISES UNIQUE MICROCODE.
```

```
                                ;
                                ;*******************************************************************************
                                ;TEST 125        TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE
                                ;*******************************************************************************
        007166                  TS125:
        007166  005000                  CLR     R0                      ;R0=0
        007170  005010                  CLR     (R0)                    ;LOC 0=0
        007172  105110                  COMB    (R0)                    ;LOC 0=177400
        007174  005110                  COM     (R0)
        007176  005004                  CLR     R4                      ;R4=0
        007200  005104                  COM     R4                      ;R4=177777
        007202  111004                  MOVB    (R0),R4                 ;R4=0
        007204  005704                  TST     R4                      ;CHECK SIGN OF WORD
        007206  001401                  BEQ     DOP1
        007210  104000                  EMT                             ;MOVB SHOULD SIGN X-TEND
        007212  005110          DOP1:   COM     (R0)                    ;LOC 0=177777
        007214  111004                  MOVB    (R0),R4                 ;DO MOVB W/ EVEN BYTE
        007216  100401                  BMI     TS126
        007220  104000                  EMT                             ;MOVB SHOULD SIGN X-TEND

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS WHICH REFERENCE
                                ;ODD BYTES.  LOC. 0 IS SET TO 177400.  R0 IS SET TO 0 AND R4 IS
                                ;SET TO 1.  THE BISB INSTRUCTION USES THE DATA IN BYTE 1 TO SET BYTE 0.
                                ;THE RESULT IS CHECKED BY INCREMENTING THE WORD (LOC. 0) TO ZERO.
                                ;
                                ;*******************************************************************************
                                ;TEST 126        TEST MODE 1-ODD BYTE W/ DOP INSTS.
                                ;*******************************************************************************
        007222                  TS126:
        007222  005000                  CLR     R0                      ;R0=0
        007224  005010                  CLR     (R0)                    ;LOC. 0=0
        007226  005004                  CLR     R4                      ;R4=0
        007230  005204                  INC     R4                      ;R4=1
        007232  105114                  COMB    (R4)                    ;LOC. 0=177400
        007234  151410                  BISB    (R4),(R0)               ;TRY TO BIS LOW ORDER BITS W/ MODE 1
        007236  005210                  INC     (R0)                    ;CHECK RESULT
        007240  001401                  BEQ     TS127
        007242  104000                  EMT                             ;RESULT OF BISB INCORRECT

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 2 DOP INSTRUCTIONS.  LOC. 0 IS SET TO -1.
                                ;R0 IS CLEARED AND USED AS THE MODE 2 ADDRESSING REGISTER TO MOVE LOC. 0
                                ;TO R7.  THE DATA RESULTS ARE VERIFIED AND THE INCREMENTING OF THE REGISTER
                                ;IS CHECKED.
                                ;
                                ;*******************************************************************************
                                ;TEST 127        TEST MODE 2 W/ DOP INSTS.
                                ;*******************************************************************************
        007244                  TS127:
        007244  005000                  CLR     R0                      ;R0=0
        007246  005010                  CLR     (R0)                    ;LOC. 0=0
        007250  005110                  COM     (R0)                    ;LOC. 0=177777
        007252  012004                  MOV     (R0)+,R4                ;TRY MOVE MODE 2,0
```

```
007254  005204                  INC     R4              ;CHECK R4
007256  001401                  BEQ     DOP2
007260  104000                  EMT                     ;RESULT OF MOV INST INCORRECT
007262  005300          DOP2:   DEC     R0              ;TEST R0 AFTER MODE 2
007264  005300                  DEC     R0
007266  001401                  BEQ     TS130
007270  104000                  EMT                     ;REGISTER NOT INCREMENTED IN MODE 2

                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH ADDRESS
                        ;EVEN BYTES.  LOC. 0 IS SET TO -1.  R0 IS CLEARED AND USED AS THE
                        ;ADDRESSING REGISTER IN A TEST WHICH TRIES TO CLEAR BYTE 1 USING
                        ;BYTE 0 DATA AND A BICB.  UNIQUE IN THIS TEST IS USE OF THE
                        ;SAME ADDRESSING REGISTER FOR BOTH SOURCE AND DESTINATION.  THE SOURCE AND
                        ;DESTINATION IS CHECKED TO INSURE PROPER FUNCTIONING.
                        ;
                        ;*****************************************************************************
                        ;TEST 130       TEST MODE 2 - EVEN BYTE W/ DOP INST.
                        ;*****************************************************************************
007272                  TS130:
007272  005000                  CLR     R0              ;R0=0
007274  010010                  MOV     R0,(R0)         ;LOC. 0=0
007276  005110                  COM     (R0)            ;LOC. 0=177777
007300  142010                  BICB    (R0)+,(R0)      ;TRY TO CLEAR BYTE 1 FROM BYTE 0 W/ BICB
007302  105737  000001          TSTB    @#1             ;CHECK RESULT
007306  001401                  BEQ     DOPB2A
007310  104000                  EMT                     ;BICB DESTINATION INCORRECT
007312  105137  000000  DOPB2A: COMB    @#0             ;CHECK BICB SOURCE
007316  001401                  BEQ     TS131
007320  104000                  EMT                     ;BICB SOURCE INCORRECTLY CHANGED
                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH REFERENCE
                        ;ODD BYTES.  R0 IS SET TO 1, LOC. 0 IS SET TO 177400, AND R4 IS CLEARED.
                        ;A MODE 2 MOVB USES R0 TO MOVE BYTE 1 TO R4.  AN INCREMENT
                        ;IS USED TO CHECK THAT THE PROPER BYTE WAS MOVED AND SIGN X-TENDED.
                        ;
                        ;*****************************************************************************
                        ;TEST 131       TEST MODE 2 - ODD BYTE W/ DOP INST.
                        ;*****************************************************************************
007322                  TS131:
007322  005000                  CLR     R0              ;R0=0
007324  005004                  CLR     R4              ;R4=0
007326  005010                  CLR     (R0)            ;LOC. 0=0
007330  005110                  COM     (R0)            ;LOC. 0=177777
007332  105120                  COMB    (R0)+           ;LOC 0=177400; R0=1
007334  112004                  MOVB    (R0)+,R4        ;TRY DOP MODE 2 W/ ODD BYTE
007336  005204                  INC     R4              ;CHECK RESULT OF MOVB
007340  001401                  BEQ     DOPB2B
007342  104000                  EMT                     ;RESULT OF MOVB INCORRECT
007344  005740          DOPB2B: TST     -(R0)           ;BUMP R0 DOWN BY 2
007346  005700                  TST     R0              ;CHECK R0
007350  001401                  BEQ     TS132
007352  104000                  EMT                     ;MODE 2 BYTE DID NOT INCREMENT REG. CORRECTLY
                        ;*****************************************************************************
```

```
                                          ;
                                          ;       THIS TEST VERIFIES MODE 3 DOUBLE-OPERAND INSTRUCTIONS.
                                          ;LOC. 0 IS LOADED WITH ALTERNATING ZEROES AND ONES; AND R0 IS LOADED
                                          ;WITH ALTERNATING ONES AND ZEROES.  A MODE 3 BIS IS USED TO SET R0
                                          ;TO -1 BY USING LOC. 0 AS THE SOURCE TO BIS THE ZEROES IN R0.  THE
                                          ;RESULT IS TESTED BY INCREMENTING R0 AND CHECKING FOR ZERO.
                                          ;
                                          ;*************************************************************************
                                          ;TEST 132       TEST MODE 3 W/ DOP INSTS.
                                          ;*************************************************************************
007354                                    TS132:
007354  012737  052525  000000                    MOV     #052525,@#0      ;MOVE 52525 TO LOC. 0
007362  012700  125252                            MOV     #125252,R0       ;SET ALT. ONE AND ZERO IN R0
007366  053700  000000                            BIS     @#0,R0           ;TRY TO SET ALL OTHER BITS W/ MODE 3
007372  005200                                    INC     R0               ;TEST RESULT
007374  001401                                    BEQ     TS133
007376  104000                                    EMT                      ;BIS W/ MODE 3 INCORRECT RESULT
                                          ;*************************************************************************
                                          ;
                                          ;       THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS WHICH
                                          ;ADDRESS EVEN BYTES.  BYTE 0 IS SET TO ALTERNATING 1'S AND 0'S; BYTE 1,
                                          ;ALTERNATING 0'S AND 1'S.  R0 IS CLEARED AND A BISB IS USED TO
                                          ;SET THE LOW BYTE OF R0 TO 252.
                                          ;
                                          ;*************************************************************************
                                          ;TEST 133       TEST MODE 3 - EVEN BYTE W/ DOP INSTS.
                                          ;*************************************************************************
007400                                    TS133:
007400  012737  052652  000000                    MOV     #52652,@#0       ;NOVE 1'S AND 0' PATTERN TO LOC. 0
007406  005000                                    CLR     R0               ;R0=0
007410  153700  000000                            BISB    @#0,R0           ;TRY R0=252 W/ MODE 3 - EVEN BYTE
007414  022700  000252                            CMP     #252,R0          ;BISB W/ EVEN BYTE SUCCESSFUL?
007420  001401                                    BEQ     TS134
007422  104000                                    EMT                      ;BISB W/ MODE 3 - EVEN BYTE FAILED
                                          ;*************************************************************************
                                          ;
                                          ;       THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS
                                          ;WHICH ADDRESS ODD BYTES.  THE SAME PROCEDURE USED IN PREVIOUS
                                          ;TEST IS USED HERE.  THIS TIME BYTE 1 IS USED AS THE SOURCE BYTE.
                                          ;THE EXPECTED RESULT IS:  R0 = 125.
                                          ;
                                          ;*************************************************************************
                                          ;TEST 134       TEST MODE 3 - ODD BYTE W/ DOP INSTS.
                                          ;*************************************************************************
007424                                    TS134:
007424  012737  052652  000000                    MOV     #52652,@#0       ;MOVE 1'S AND 0'S PATTERN TO LOC 0
007432  005000                                    CLR     R0               ;R0=0
007434  153700  000001                            BISB    @#1,R0           ;TRY R0=152 W/ MODE 3 - ODD BYTE
007440  022700  000125                            CMP     #125,R0          ;R0=125?
007444  001401                                    BEQ     TS135
007446  104000                                    EMT                      ;BISB W/ MODE 3 - ODD BYTE FAILED

                                          ;*************************************************************************
                                          ;TEST 135       TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING MST
                                          ;*************************************************************************
007450                                    TS135:
```

```
007450  005000                          CLR     R0                      ;R0=0
007452  105100                          COMB    R0                      ;R0=377
007454  000263                          +SEC!SEV                        ;SET C AND V BITS
007456  132700  000200                  BITB    #200,R0                 ;TRY DOPNM DEST. MODE 0-BYTE
007462  001403                          BEQ     DNMB0A                  ;BR TO ERROR IF Z BIT SET
007464  102402                          BVS     DNMB0A                  ;BR TO ERROR IF V BIT SET
007466  103001                          BCC     DNMB0A                  ;BR TO ERROR IF C BIT CLEAR.
007470  100401                          BMI     DNMB0B
007472                          DNMB0A:
007472  104000                          EMT                             ;CC'S INCORRECT
007474  105100                  DNMB0B: COMB    R0                      ;CHECK DESTINATION DATA
007476  001401                          BEQ     TS136
007500  104000                          EMT                             ;DEST. DATA MODIFIED

                                ;*******************************************************************************
                                ;TEST 136        TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST
                                ;*******************************************************************************
007502                          TS136:
007502  005000                          CLR     R0                      ;R0=0
007504  005010                          CLR     (R0)                    ;LOC. 0=0
007506  000241                          CLC                             ;CLEAR C BIT
007510  032710  177777                  BIT     #177777,(R0)            ;TRY DOPNM DEST. MODE 1
007514  100403                          BMI     DNM1A                   ;BR TO ERROR IF N BIT SET
007516  102402                          BVS     DNM1A                   ;BR TO ERROR IF V BIT SET
007520  103401                          BCS     DNM1A                   ;BR TO ERROR IF C BIT SET
007522  001401                          BEQ     DNM1B
007524                          DNM1A:
007524  104000                          EMT                             ;COND. CODES INCORRECT
007526  005710                  DNM1B:  TST     (R0)                    ;CHECK TEST DATA
007530  001401                          BEQ     TS137
007532  104000                          EMT                             ;DESTINATION DATA MODIFIED

                                ;*******************************************************************************
                                ;TEST 137        TEST DEST, MODE 2 W/ DOP NON-MODIFYING INST.
                                ;*******************************************************************************
007534                          TS137:
007534  005000                          CLR     R0                      ;R0=0
007536  005010                          CLR     (R0)                    ;LOC. 0=0
007540  052710  125252                  BIS     #125252,(R0)            ;LOC. 0=125252
007544  032720  077777                  BIT     #77777,(R0)+            ;TRY DOPNM INST W/ MODE 2
007550  102402                          BVS     DNM2A                   ;BR TO ERROR IF V BIT SET
007552  001401                          BEQ     DNM2A                   ;BR TO ERROR IF Z-BIT SET
007554  100001                          BPL     DNM2B
007556                          DNM2A:
007556  104000                          EMT                             ;COND. CODES INCORRECT
007560  005300                  DNM2B:  DEC     R0                      ;DECREMENT R0 TO CHECK IT.
007562  005300                          DEC     R0
007564  001401                          BEQ     DNM2D
007566                          DNM2C:
007566  104000                          EMT                             ;MODE 2 REGISTER NOT INCREMENTED BY 2
007570  022710  125252          DNM2D:  CMP     #125252,(R0)            ;CHECK DEST. DATA
007574  001401                          BEQ     TS140
007576  104000                          EMT                             ;DEST. DATA MODIFIED

                                ;*******************************************************************************
                                ;TEST 140        TEST DEST. MODE 2-BYTE, W/DOP NON-MODIFYING INST
```

```
                                 ;*****************************************************************************
007600                           TS140:
007600  005000                           CLR     R0                      ;R0=0
007602  005010                           CLR     (R0)                    ;LOC. 0=0
007604  052710  052652                   BIS     #52652,(R0)             ;LOC. 0=52652
007610  000263                           +SEC!SEV                        ;SET C AND V BITS
007612  132720  000201                   BITB    #201,(R0)+              ;TRY DOPNM INST. W/ MODE 2 EVEN BYTE
007616  001403                           BEQ     DNMB2A                  ;BR TO ERROR IF Z-BIT SET
007620  103002                           BCC     DNMB2A                  ;BR TO ERROR IF C-BIT CLEAR
007622  102401                           BVS     DNMB2A                  ;BR TO ERROR IF V-BIT SET
007624  100401                           BMI     DNMB2B
007626                           DNMB2A:
007626  104000                           EMT                             ;COND. CODES INCORRECT
007630  005300                   DNMB2B: DEC     R0                      ;CHECK DEST. REGISTER.
007632  001401                           BEQ     DNMB2C
007634  104000                           EMT                             ;DEST. REGISTER NOT INCREMENTED BY 1
007636  005200                   DNMB2C: INC     R0                      ;R0=1
007640  132720  000201                   BITB    #201,(R0)+              ;TRY DOPNM INST. W/MODE 2-ODD BYTE
007644  001402                           BEQ     DNMB2D  ;BR TO ERROR IF Z-BIT SET
007646  102401                           BVS     DNMB2D                  ;BR TO ERROR IF V-BIT SET
007650  100001                           BPL     DNMB2E
007652                           DNMB2D:
007652  104000                           EMT                             ;COND. CODES INCORRECT
007654  005300                   DNMB2E: DEC     R0                      ;DEC R0 TO CHECK IT.
007656  005300                           DEC     R0
007660  001401                           BEQ     DNMB2F
007662  104000                           EMT                             ;DEST. REGISTER NOT INCREMENTED BY 1
007664  022710  052652           DNMB2F: CMP     #52652,(R0)             ;CHECK DEST. DATA IS UNMODIFIED
007670  001401                           BEQ     TS141
007672  104000                           EMT                             ;DEST. DATA WAS MODIFIED.


                                 ;*****************************************************************************
                                 ;TEST 141       TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.
                                 ;*****************************************************************************
007674                           TS141:
007674  005000                           CLR     R0                      ;R0=0
007676  005010                           CLR     (R0)                    ;LOC. 0=0
007700  052710  125125                   BIS     #125125,(R0)            ;LOC. 0=125125
007704  105100                           COMB    R0                      ;R0=377
007706  005200                           INC     R0                      ;R0=400
007710  005010                           CLR     (R0)                    ;LOC. 400=0
007712  000263                           +SEC!SEV                        ;C-BIT=V-BIT=1
007714  132730  000201                   BITB    #201,@(R0)+             ;TRY DOPNM W/MODE 3-EVEN BYTE
007720  001403                           BEQ     DNMB3A                  ;BR TO ERROR IF Z BIT SET
007722  102402                           BVS     DNMB3A                  ;BR TO ERROR IF V BIT SET
007724  103001                           BCC     DNMB3A                  ;BR TO ERROR IF C BIT CLEAR
007726  100001                           BPL     DNMB3B
007730                           DNMB3A:
007730  104000                           EMT                             ;COND. CODES INCORRECT
007732  022700  000402           DNMB3B: CMP     #402,R0                 ;CHECK DEST. REGISTER INC. BY 2 AND INC BY 2 AGAIN
007736  001401                           BEQ     DNMB3C
007740  104000                           EMT                             ;DEST. REGISTER NOT INCREMENTED BY 2
007742  005200                   DNMB3C: INC     R0                      ;R0=404
007744  005200                           INC     R0
007746  132730  000201                   BITB    #201,@(R0)+             ;TRY DOPNM DEST MODE 3-BYTE(ODD)
```

```
007752  001402                          BEQ     DNMB3D          ;BR TO ERROR IF Z BIT SET
007754  102401                          BVS     DNMB3D          ;BR TO ERROR IF V BIT SET
007756  100401                          BMI     DNMB3E
007760                          DNMB3D:
007760  104000                          EMT                     ;COND. CODES INCORRECT
007762  005004                  DNMB3E: CLR     R4              ;R4=0
007764  022714  125125                  CMP     #125125,(R4)    ;CHECK DEST. DATA
007770  001401                          BEQ     TS142
007772  104000                          EMT                     ;DEST. DATA MODIFIED

                                ;*************************************************************************
                                ;TEST 142       TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.
                                ;*************************************************************************
007774                          TS142:
007774  005000                          CLR     R0              ;R0=0
007776  005010                          CLR     (R0)            ;LOC. 0=0
010000  052710  125252                  BIS     #125252,(R0)    ;LOC. 0=125125
010004  052700  000002                  BIS     #2,R0           ;R0=2
010010  000277                          SCC                     ;SET ALL COND. CODE BITS
010012  032740  020000                  BIT     #20000,-(R0)    ;TRY DOPNM W/ MODE 4
010016  100403                          BMI     DNM4A           ;BR TO ERROR IF N-BIT SET
010020  102402                          BVS     DNM4A           ;BR TO ERROR IF V-BIT SET
010022  103001                          BCC     DNM4A           ;BR TO ERROR IF C-BIT CHAR
010024  001001                          BNE     DNM4B
010026                          DNM4A:
010026  104000                          EMT                     ;COND. CODES INCORRECT
010030  005700                  DNM4B:  TST     R0              ;CHECK DEST. REGISTER
010032  001401                          BEQ     DNM4C
010034  104000                          EMT                     ;DEST. REGISTER NOT DECREMENTED BY 2
010036  022737  125252  000000  DNM4C:  CMP     #125252,@#0     ;CHECK DEST. DATA
010044  001401                          BEQ     TS143
010046  104000                          EMT                     ;DEST. DATA MODIFIED

                                ;*************************************************************************
                                ;TEST 143       TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
                                ;*************************************************************************
010050                          TS143:
010050  005000                          CLR     R0              ;R0=0
010052  005010                          CLR     (R0)            ;LOC. 0=0
010054  052710  052652                  BIS     #52652,(R0)     ;LOC. 0=52652
010060  052700  000002                  BIS     #2,R0           ;R0=2
010064  000257                          CCC                     ;COND. CODES=0
010066  132740  000201                  BITB    #201,-(R0)      ;TRY DOPNM INST W/MODE 4 ODD BYTE
010072  102403                          BVS     DNMB4A          ;BR TO ERROR IF V BIT SET
010074  001402                          BEQ     DNMB4A  ;BR TO ERROR IF Z BIT SET
010076  103401                          BCS     DNMB4A          ;BR TO ERROR IF C BIT SET
010100  001001                          BNE     DNMB4B
010102                          DNMB4A:
010102  104000                          EMT                     ;COND. CODES INCORRECT
010104  022700  000001          DNMB4B: CMP     #1,R0           ;CHECK DEST. REGISTER
010110  001401                          BEQ     DNMB4C
010112  104000                          EMT                     ;DEST REG. NOT DECREMENTED BY 1
010114  132740  000201          DNMB4C: BITB    #201,-(R0)      ;TRY DOPNM INST. W/MODE 4 EVEN BYTE
010120  001401                          BEQ     DNMB4D          ;BR TO ERROR IF Z-BIT SET
010122  100401                          BMI     DNMB4E
010124                          DNMB4D:
```

```
010124  104000                          EMT                             ;COND. CODES INCORRECT
010126  005700                  DNMB4E: TST     R0                      ;CHECK DEST. REGISTER
010130  001401                          BEQ     DNMB4F
010132  104000                          EMT                             ;DEST. REG. NOT DECREMENTED BY 1
010134  022710  052652          DNMB4F: CMP     #52652,(R0)             ;CHECK DESTINATION DATA
010140  001401                          BEQ     TS144
010142  104000                          EMT                             ;DEST. DATA MODIFIED

                                ;*************************************************************************
                                ;TEST 144       TEST DEST MODE 5 W/DOP NON-MODIFYING INST.
                                ;*************************************************************************
010144                          TS144:
010144  005000                          CLR     R0                      ;R0=0
010146  005010                          CLR     (R0)                    ;LOC 0=0
010150  052710  100000                  BIS     #100000,(R0)            ;LOC. 0=100000
010154  052700  000402                  BIS     #402,R0                 ;R0=2
010160  000277                          SCC                             ;SET ALL COND. CODE BITS
010162  032750  100000                  BIT     #100000,@-(R0)          ;TRY DOPNM W/MODE 5
010166  102403                          BVS     DNM5A                   ;BR TO ERROR IF V-BIT SET
010170  103002                          BCC     DNM5A                   ;BR TO ERROR IF C-BIT CLEAR
010172  001401                          BEQ     DNM5A                   ;BR TO ERROR IF Z-BIT SET
010174  100401                          BMI     DNM5B
010176                          DNM5A:
010176  104000                          EMT                             ;COND, CODES INCORRECT
010200  022700  000400          DNM5B:  CMP #400,R0                     ;CHECK DEST. REGISTER
010204  001401                          BEQ     DNM5C
010206  104000                          EMT                             ;DEST. REGISTER NOT DECREMENTED BY 2
010210  022737  100000  000000  DNM5C:  CMP     #100000,@#0             ;CHECK DESTINATION DATA
010216  001401                          BEQ     TS145
010220  104000                          EMT                             ;DEST. DATA INCORRECTLY MODIFIED

                                ;*************************************************************************
                                ;TEST 145       TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
                                ;*************************************************************************
010222                          TS145:
010222  005000                          CLR     R0                      ;R0=0
010224  005010                          CLR     (R0)                    ;LOC> 0=0
010226  052710  000001                  BIS     #1,(R0)                 ;LOC. 0=1
010232  005100                          COM     R0                      ;R0=-1 C-BIT=1
010234  032760  000001  000001          BIT     #1,1(R0)                ;TRY DOPNM W/MODE 6
010242  001403                          BEQ     DNM6A                   ;BR TO ERROR IF Z-BIT SET
010244  102402                          BVS     DNM6A                   ;BR TO ERROR IF V-BIT SET
010246  103001                          BCC     DNM6A                   ;BR TO ERROR IF C-BIT CLEAR
010250  100001                          BPL     DNM6B
010252                          DNM6A:
010252  104000                          EMT                             ;COND CODES INCORRECT
010254  022700  177777          DNM6B:  CMP     #-1,R0                  ;CHECK DEST. REGISTER
010260  001401                          BEQ     DNM6C
010262  104000                          EMT                             ;DEST. REGISTER MODIFIED
010264  022737  000001  000000  DNM6C:  CMP     #1,@#0                  ;CHECK DEST. DATA
010272  001401                          BEQ     TS146
010274  104000                          EMT                             ;DEST. DATA MODIFIED

                                ;*************************************************************************
                                ;TEST 146       TEST DEST MODE 7 W/DOP NON-MODIFYING INST.
                                ;*************************************************************************
```

```
010276                                  TS146:
010276  005000                                  CLR     R0              ;R0=0
010300  005010                                  CLR     (R0)            ;LOC. 0=0 C-BIT=0
010302  052710  125125                          BIS     #125125,(R0)    ;LOC. 0=125125
010306  052700  000001                          BIS     #1,R0           ;R0=1
010312  132770  000125  000403                  BITB    #125,@403(R0)   ;TRY DOPNM W/MODE 7
010320  102403                                  BVS     DNM7A           ;BR TO ERROR IF V-BIT SET
010322  100402                                  BMI     DNM7A           ;BR TO ERROR IF N-BIT SET
010324  103401                                  BCS     DNM7A           ;BR TO ERROR IF C-BIT SET
010326  001401                                  BEQ     DNM7B
010330                                  DNM7A:
010330  104000                                  EMT                     ;COND. CODES INCORRECT
010332  022700  000001                  DNM7B:  CMP     #1,R0           ;CHECK DEST. REGISTER
010336  001401                                  BEQ     DNM7C
010340  104000                                  EMT                     ;DESTINATION REGISTER MODIFIED
010342  022737  125125  000000          DNM7C:  CMP     #125125,@#0     ;CHECK DEST. DATA
010350  001401                                  BEQ     TS147
010352  104000                                  EMT                     ;DEST. DATA INCORRECT

                                        ;*****************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 1 INSTRUCTION.
                                        ;DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED TO LOC. 0
                                        ;USING MOV SRC MODE 0, DEST. MODE 1.
                                        ;
                                        ;*****************************************************************************
                                        ;TEST 147       TEST MOV DESTINATION MODE 1
                                        ;*****************************************************************************
010354                                  TS147:
010354  005000                                  CLR     R0              ;R0=0
010356  005010                                  CLR     (R0)            ;LOC. 0=0
010360  005100                                  COM     R0              ;R0=-1
010362  005004                                  CLR     R4              ;R4 POINTS TO LOC. 0
010364  010014                                  MOV     R0,(R4)         ;TRY MOVE MODE 0,1
010366  102402                                  BVS     MDM1A           ;BR TO ERROR IF V SET
010370  001401                                  BEQ     MDM1A           ;BR TO ERROR IF Z SET
010372  100401                                  BMI     MDM1B
010374                                  MDM1A:
010374  104000                                  EMT                     ;CONDITION CODE NOT CORRECT
010376  005704                          MDM1B:  TST     R4
010400  001401                                  BEQ     TS150
010402  104000                                  EMT                     ;DESTINATION REGISTER INCORRECTLY ALTERED

                                        ;*****************************************************************************
                                        ;
                                        ;
                                        ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 2 INSTRUCTION.
                                        ;       DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED
                                        ;       TO LOCATION 0 USING MOV SRC MODE 0, DEST. MODE 1.
                                        ;
                                        ;*****************************************************************************
                                        ;TEST 150       TEST MOV DESTINATION MODE 2
                                        ;*****************************************************************************
010404                                  TS150:
010404  005000                                  CLR     R0              ;R0=0
010406  005001                                  CLR     R1              ;R1=0
010410  005010                                  CLR     (R0)            ;LOC.0=0
```

```
010412  005110                          COM     (R0)            ;LOC. 0=-1
010414  010120                          MOV     R1,(R0)+        ;TRY MOVE MODE 0,2
010416  100402                          BMI     MDM2A           ;BR TO ERROR IF N SET
010420  102401                          BVS     MDM2A           ;BR TO ERROR IF V SET
010422  001401                          BEQ     MDM2B
010424                          MDM2A:
010424  104000                          EMT                     ;CC'S INCORRECT
010426  005300                  MDM2B:  DEC     R0
010430  005300                          DEC     R0
010432  001401                          BEQ     MDM2D
010434                          MDM2C:
010434  104000                          EMT                     ;DESTINATION REGISTER NOT INCREMENTED PROPERLY
010436  005737  000000          MDM2D:  TST     @#0
010442  001401                          BEQ     TS151
010444  104000                          EMT                     ;DESTINATION DATA INCORRECT

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES DESTINATION MODE 2 W/MOVB INSTS. TWO DIFFERENT MOVB
                                ;INSTRUCTIONS ARE USED TO MOVE A TEST PATTERN FIRST TO BYTE 0 THEN TO BYTE 1.
                                ;
                                ;*******************************************************************************
                                ;TEST 151       TEST MOV-BYTE DESTINATION MODE 2
                                ;*******************************************************************************
010446                          TS151:
010446  005000                          CLR     R0              ;R0=0
010450  005010                          CLR     (R0)            ;LOC. 0=0
010452  112720  000125                  MOVB    #125,(R0)+      ;TRY DESTINATION MODE 2 W/EVEN BYTE
010456  102402                          BVS     MBDM2A          ;BR TO ERROR IF V SET
010460  001401                          BEQ     MBDM2A          ;BR TO ERROR IF Z SET
010462  100001                          BPL     MBDM2B
010464                          MBDM2A:
010464  104000                          EMT                     ;CC'S INCORRECT
010466  022700  000001          MBDM2B: CMP     #1,R0
010472  001401                          BEQ     MBDM2C
010474  104000                          EMT                     ;REGISTER NOT INCREMENTED BY ONE
010476  112720  000252          MBDM2C: MOVB    #252,(R0)+      ;TRY DESTINATION MODE 2 W/ODD BYTE
010502  102402                          BVS     MBDM2D
010504  001401                          BEQ     MBDM2D
010506  100401                          BMI     MBDM2E
010510                          MBDM2D:
010510  104000                          EMT                     ;CC'S NOT SET CORRECT
010512  022700  000002          MBDM2E: CMP     #2,R0
010516  001401                          BEQ     MBDM2F
010520  104000                          EMT                     ;REGISTER NOT INCREMENTED BY ONE
010522  022737  125125  000000  MBDM2F: CMP     #125125,@#0     ;CHECK DATA
010530  001401                          BEQ     TS152
010532  104000                          EMT                     ;DESTINATION DATA INCORRECT

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MOV DESTINATION MODE 3. R0 IS USED TO PICK UP
                                ;AN ADDRESS AT LOC. 400. LOC 400 POINTS TO LOC. 0 THE EFFECTIVE DEST. ADDR.. ALSO, MOVB
                                ;INST. ARE USED W/ EVEN AND ODD BYTES TO CHECK MOV BYTES INST AND MODE 37 DESTINATIONS.
                                ;*******************************************************************************
                                ;TEST 152       TEST MOV(B) DESTINATION MODE 3
```

```
                                        ;*****************************************************************************
010534                                  TS152:
010534  012700  000400                          MOV     #400,R0         ;R0=400
010540  005010                                  CLR     (R0)            ;LOC. 400 POINTS TO LOC. 0
010542  005037  000000                          CLR     @#0             ;LOC. 0=0
010546  012730  125252                          MOV     #125252,@(R0)+  ;TRY MOV DESTINATION MODE 2
010552  102402                                  BVS     MDM3A           ;BR TO ERROR IF V SET
010554  001401                                  BEQ     MDM3A           ;BR TO ERROR IF Z SET
010556  100401                                  BMI     MDM3B
010560                                  MDM3A:
010560  104000                                  EMT                     ;CC'S INCORRECT
010562  022700  000402                  MDM3B:  CMP     #402,R0         ;CHECK DEST. MODE REGISTER
010566  001401                                  BEQ     MDM3C
010570  104000                                  EMT                     ;REGISTER NOT INCREMENTED BY 2
010572  022737  125252  000000          MDM3C:  CMP     #125252,@#0     ;CHECK DESTINATION DATA
010600  001401                                  BEQ     MDM3D
010602  104000                                  EMT                     ;DESTINATION DATA INCORRECT
010604  112737  000125  000000          MDM3D:  MOVB    #125,@#0        ;TRY MOVB DESTINATION MODE Z EVEN BYTE
010612  022737  125125  000000                  CMP     #125125,@#0     ;CHECK DATA
010620  001401                                  BEQ     MDM3E
010622  104000                                  EMT                     ;DESTINATION DATA INCORRECT
010624  112737  000525  000001          MDM3E:  MOVB    #525,@#1        ;TRY MOVB DESTINATION MODE 2 ODD BYTE
010632  022737  052525  000000                  CMP     #52525,@#0      ;CHECK DATA
010640  001401                                  BEQ     TS153
010642  104000                                  EMT                     ;

                                        ;*****************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 4 INSTRUCTION.
                                        ;SOP INSTRUCTIONS ON R0 ARE USED TO CLEAR TARGET LOCATION 0.
                                        ;R4 IS USED AS THE MODE 4 ADDRESSING REGISTER, AND
                                        ;CONDITIONAL BRANCHES ARE USED TO VERIFY THE DATA.
                                        ;
                                        ;*****************************************************************************
                                        ;TEST 153       TEST MOV DESTINATION MODE 4
                                        ;*****************************************************************************
010644                                  TS153:
010644  005000                                  CLR     R0              ;R0=0
010646  005010                                  CLR     (R0)            ;LOC 0=0
010650  012704  000002                          MOV     #2,R4           ;R4=2
010654  012744  012345                          MOV     #12345,-(R4)    ;TRY MOV DEST. MODE 4
010660  102402                                  BVS     MDM4A           ;BR TO ERROR IF V-BIT SET
010662  001401                                  BEQ     MDM4A           ;BR TO ERROR IF Z-BIT SET
010664  100001                                  BPL     MDM4B
010666                                  MDM4A:
010666  104000                                  EMT                     ;CC'S NOT CORRECT
010670  005704                          MDM4B:  TST     R4              ;CHECK DECREMENTING OF MODE 4 REG.
010672  001401                                  BEQ     MDM4C
010674  104000                                  EMT                     ;DESTINATION MODE REGISTER NOT DECREMENTED BY 2
010676  022710  012345                  MDM4C:  CMP     #12345,(R0)     ;CHECK DESTINATION DATA
010702  001401                                  BEQ     TS154
010704  104000                                  EMT                     ;DESTINATION DATA INCORRECT

                                        ;*****************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES THE MOVB DESTINATION MODE 4 INSTRUCTION
```

```
                                    ;       ON BOTH ODD AND EVEN BYTES. SOP INSTRUCTIONS ON R4 ARE
                                    ;       USED TO CLEAR TARGET LOCATION 0. R0 IS USED AS THE MODE 4
                                    ;       ADDRESSING REGISTER, AND CMP AND CONDITIONAL BRANCH
                                    ;       INSTRUCTIONS ARE USED TO VERIFY THE DATA.

                                    ;*************************************************************************
                                    ;TEST 154       TEST MOVB DESTINATION MODE 4
                                    ;*************************************************************************
010706                              TS154:
010706  005004                              CLR     R4              ;R4=0
010710  005014                              CLR     (R4)            ;LOC. 0=0
010712  012700  000002                      MOV     #2,R0           ;R0 = 2
010716  112740  125125                      MOVB    #125125,-(R0)   ;TRY MOVB DEST. MODE 4-ODD BYTE
010722  020027  000001                      CMP     R0,#1           ;CHECK THAT DEST. REG. WAS DECREMENTED
010726  001401                              BEQ     MBDM4A
010730  104000                              EMT                     ;DESTINATION REG. NOT DECREMENTED BY 1
010732  021427  052400              MBDM4A: CMP     (R4),#52400     ;CHECK DEST. DATA
010736  001401                              BEQ     MBDM4B
010740  104000                              EMT                     ;DEST. DATA NOT CORRECT
010742  112740  125125              MBDM4B: MOVB    #125125,-(R0)   ;TRY MOVB DEST. MODE 4--EVEN BYTE
010746  102402                              BVS     MBDM4C          ;BR. TO ERROR IF V-BIT SET
010750  001401                              BEQ     MBDM4C          ;BR TO ERROR IF Z-BIT SET
010752  100001                              BPL     MBDM4D
010754                              MBDM4C:
010754  104000                              EMT                     ;COND. CODES INCORRECT
010756  005700                      MBDM4D: TST     R0              ;CHECK MODE 4 DEST. REGISTER
010760  001401                              BEQ     MBDM4E
010762  104000                              EMT                     ;DESTINATION REG NOT DECREMENTED BY 1
010764  021427  052525              MBDM4E: CMP     (R4),#52525     ;CHECK DEST. DAT^A
010770  001401                              BEQ     TS155
010772  104000                              EMT                     ;DESTINATION DATA INCORRECT

                                    ;*************************************************************************
                                    ;
                                    ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 5 AND THE MOVB
                                    ;DESTINATION MODE 5 - EVEN BYTE INSTRUCTIONS. R4 IS A
                                    ;POINTER TO TARGET LOCATION 0 AND R0 IS SETUP TO
                                    ;POINT TO LOCATION 376 FOR THE MOV, AND LOCATION 404 FOR
                                    ;THE MOVB INSTRUCTIONS. CMP INSTRUCTIONS ARE USED TO VERIFY
                                    ;PROPER ADDRESSING AND DATA.

                                    ;*************************************************************************
                                    ;TEST 155       TEST MOV DESTINATION MODE 5
                                    ;*************************************************************************
010774                              TS155:
010774  005004                              CLR     R4              ;R4=0
010776  005014                              CLR     (R4)            ;LOC. 0 = 0
011000  012700  000400                      MOV     #400,R0         ;R0=400
011004  012750  004321                      MOV     #4321,@-(R0)    ;TRY MOV DEST. MODE 5
011010  102402                              BVS     MDM5A           ;BR TO ERROR IF V-BIT SET
011012  001401                              BEQ     MDM5A           ;BR TO ERROR IF Z-BIT SET
011014  100001                              BPL     MDM5B
011016                              MDM5A:
011016  104000                              EMT                     ;COND. CODES INCORRECT
011020  022700  000376              MDM5B:  CMP     #376,R0         ;CHECK MODE 5 REG. WAS DECREMENTED
011024  001401                              BEQ     MDM5C
```

```
011026  104000                         EMT                          ;MODE 5 REGISTER NOT DECREMENTED BY 2
011030  022714  004321         MDM5C:  CMP     #4321,(R4)           ;CHECK DEST. DATA
011034  001401                         BEQ     MDM5D
011036  104000                         EMT                          ;DEST. DATA INCORRECT
011040  012700  000406         MDM5D:  MOV     #406,R0              ;R0=406
011044  112750  000377                 MOVB    #377,@-(R0)          ;TRY MOV DEST. MODE 5 --EVEN BYTE
011050  022700  000404                 CMP     #404,R0              ;CHECK MODE 5 REG.
011054  001401                         BEQ     MDM5E
011056  104000                         EMT                          ;MODE 5 REGISTER NOT DECREMENTED BY 2
011060  022714  177721         MDM5E:  CMP     #177721,(R4)         ;CHECK DEST. DATA
011064  001401                         BEQ     TS156
011066  104000                         EMT                          ;DEST. DATA INCORRECT

                               ;*******************************************************************************
                               ;
                               ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 6 AND MOVB - EVEN BYTE
                               ;DESTINATION MODE 6 INDTRUCTIONS. R0 IS USED TO SETUP TARGET LOC.0
                               ;FOR BOTH TESTS. PATTERNS OF ONES AND ZEROES ARE MOVED INTO LOC.0
                               ;BY MODE 6 INSTRUCTIONS, AND CMP INSTRUCTIONS ARE USED TO VERIFY
                               ;PROPER ADDRESSING AND DATA.
                               ;
                               ;*******************************************************************************
                               ;TEST 156       TEST MOV DESTINATION MODE 6
                               ;*******************************************************************************
011070                         TS156:
011070  005000                         CLR     R0                   ;R0=0
011072  005010                         CLR     (R0)                 ;LOC. 0=0
011074  005200                         INC     R0                   ;R0=1
011076  012760  052525  177777         MOV     #052525,-1(R0)       ;TRY MOV DEST. MODE 6
011104  102402                         BVS     MDM6A                ;BR TO ERROR IF V-BIT SET
011106  001401                         BEQ     MDM6A                ;BR TO ERROR IF Z-BIT SET
011110  100001                         BPL     MDM6B
011112                         MDM6A:
011112  104000                         EMT                          ;COND. CODES INCORRECT
011114  022700  000001         MDM6B:  CMP     #1,R0                ;CHECK DEST. REGISTER UNALTERED
011120  001401                         BEQ     MDM6C
011122  104000                         EMT                          ;DEST. REGISTER INCORRECTLY ALTERED
011124  022737  052525  000000 MDM6C:  CMP     #52525,@#0           ;CHECK DEST. DATA
011132  001401                         BEQ     MDM6D
011134  104000                         EMT                          ;DEST. DATA INCORRECT
011136  012700  000002         MDM6D:  MOV     #2,R0                ;R0=2
011142  112760  000377  177777         MOVB    #377,-1(R0)          ;TRY MOVB DEST. MODE 6
011150  022700  000002                 CMP     #2,R0                ;CHECK DEST. REGISTER UNALTERED
011154  001401                         BEQ     MDM6E
011156  104000                         EMT                          ;DEST. REGISTER INCORRECTLY ALTERED
011160  022737  177525  000000 MDM6E:  CMP     #177525,@#0          ;CHECK DEST. DATA
011166  001401                         BEQ     TS157
011170  104000                         EMT                          ;DEST. DATA INCORRECT

                               ;*******************************************************************************
                               ;
                               ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 7 AND MOVB - ODD BYTE
                               ;DESTINATION MODE 7 INSTRUCTIONS. R4 POINTS TO TARGET LOC.0 AND R0
                               ;IS USED AS THE MODE 7 ADDRESSING REGISTER. CMP INSTRUCTIONS ARE
                               ;USED TO VERIFY PROPER ADDRESSING AND DATA.
                               ;
```

```
                                      ;*******************************************************************************
                                      ;TEST 157        TEST MOV DESTINATION MODE 7
                                      ;*******************************************************************************
011172                                TS157:
011172  005004                                CLR     R4                  ;R4=0
011174  005014                                CLR     (R4)                ;LOC.0=0
011176  012700  000403                        MOV     #403,R0             ;R0=403
011202  012770  070707  177777                MOV     #70707,a-1(R0)      ;TRY MOV W/DEST MODE 7
011210  102402                                BVS     MDM7A               ;BR. TO ERROR IF V-BIT SET
011212  001401                                BEQ     MDM7A               ;BR TO ERROR IF Z-BIT SET
011214  100001                                BPL     MDM7B
011216                                MDM7A:
011216  104000                                EMT                         ;COND. CODES INCORRECT
011220  022700  000403                MDM7B:  CMP     #403,R0             ;CHECK DEST. REGISTER
011224  001401                                BEQ     MDM7C
011226  104000                                EMT                         ;DEST. REGISTER INCORRECTLY ALTERED
011230  022737  070707  000000        MDM7C:  CMP     #70707,a#0          ;CHECK DEST. DATA
011236  001401                                BEQ     MDM7D
011240  104000                                EMT                         ;DEST. DATA INCORRECT
011242  112770  107070  000001        MDM7D:  MOVB    #107070,a1(R0)      ;TRY MOVB W/DEST MODE 7--ODD BYTE
011250  022700  000403                        CMP     #403,R0             ;CHECK MODE 7 DEST. REG.
011254  001401                                BEQ     MDM7E
011256  104000                                EMT                         ;DEST. DATA INCORRECT
011260  022737  034307  000000        MDM7E:  CMP     #34307,a#0          ;CHECK DEST. DATA
011266  001401                                BEQ     TS160
011270  104000                                EMT                         ;DESTINATION DATA INCORRECT

                                      ;*******************************************************************************
                                      ;
                                      ;       THIS TEST VERIFIES MODE 4 DOUBLE OPERAND INSTRUCTIONS.
                                      ;THE TEST USES MODE 4 ADDRESSING WITH REGISTER 0 TO MOVE THRU A
                                      ;TABLE OF OPERANDS.  THE TABLE OF OPERANDS AND THE WORK LOCATION IS
                                      ;STORED FOLLOWING THE TEST CODE.  A SERIES OF 5 DOP INSTRUCTIONS UTILIZES
                                      ;THE DATA IN THE TABLE TO CYCLE THE WORK LOCATION THRU A SET OF
                                      ;VALUE.  THE DATA HAS BEEN CHOSEN TO INSURE THAT NO SINGLE ERROR WILL
                                      ;GO UNDETECTED.  WORD AND BYTE INSTRUCTION ACCESSING BOTH EVEN AND
                                      ;ODD ADDRESSES ARE USED IN THE TEST.  THE LISTING SHOWS THE
                                      ;EXPECTED INTERMEDIATE RESULT AS EACH INSTRUCTION IS EXECUTED.
                                      ;
                                      ;*******************************************************************************
                                      ;TEST 160        TEST MODE 4 W/ DOP INSTS.
                                      ;*******************************************************************************
011272                                TS160:
011272  012700  011336                        MOV     #TBL1,R0            ;INITIALIZE R0
011276  014037  011336                        MOV     -(R0),a#TBL1        ;TBL1=125252
011302  064037  011336                        ADD     -(R0),a#TBL1        ;TBL1=000377
011306  144037  011336                        BICB    -(R0),a#TBL1        ;TBL1=000252
011312  154037  011337                        BISB    -(R0),a#TBL1+1      ;TBL1=125252
011316  024037  011336                        CMP     -(R0),a#TBL1        ;CHECK RESULT
011322  001406                                BEQ     TS161
011324                                DOP4:
011324  104000                                EMT                         ;RESULT OF MODE 4 INSTS. INCORRECT

011326  125252                                125252
011330  052652                                52652
011332  053125                                53125
```

```
011334  125252                          125252
011336  000000                  TBL1:   0

                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 5 DOUBLE OPERAND INSTRUCTIONS.
                                ;THE TEST USES AN ADDRESS TABLE STORED FOLLOWING THE TEST CODE.
                                ;THIS TABLE IS SIMPLY A TABLE OF ADDRESS POINTERS WHICH ADDRESS
                                ;THE DATA TABLE USED IN THE PREVIOUS TEST.  THE TEST IS IDENTICAL TO
                                ;THE PREVIOUS TEST EXCEPT THE DATA IS REFERENCED USING THIS ADDRESS
                                ;TABLE AND MODE 5 ADDRESSING.  (SEE PREVIOUS TEST).
                                ;
                                ;*****************************************************************************
                                ;TEST 161       TEST MODE 5 W/ DOP INSTS.
                                ;*****************************************************************************
011340                          TS161:
011340  012700  011406                  MOV     #TBL2+2,R0      ;INITIALIZE R0
011344  015037  011336                  MOV     @-(R0),@#TBL1   ;TBL1=125252
011350  065037  011336                  ADD     @-(R0),@#TBL1   ;TBL1=000377
011354  145037  011336                  BICB    @-(R0),@#TBL1   ;TBL1=000252
011360  155037  011337                  BISB    @-(R0),@#TBL1+1 ;TBL1=125252
011364  025037  011336                  CMP     @-(R0),@#TBL1   ;CHECK RESULT
011370  001406                          BEQ     TS162
011372                          DOP5:
011372  104000                          EMT                     ;RESULT OF MODE 5 INSTS. INCORRECT
011374  011326                          TBL1-10
011376  011330                          TBL1-6
011400  011331                          TBL1-5
011402  011332                          TBL1-4
011404  011334                  TBL2:   TBL1-2
                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 6 DOUBLE OPERAND INSTRUCTIONS.
                                ;IT USES THE SAME DATA AS THAT USED IN THE MODE 4 TESTS.
                                ;THIS TIME THE DATA IS ACCESSED USING MODE 6.  R0 IS SET
                                ;TO POINT TO THE MIDDLE OF THE TABLE.  THE TABLE IS ACCESSED FROM
                                ;BOTTOM TO TOP BY VARYING THE OFFSET IN THE MODE 6 INSTRUCTIONS.
                                ;THE DATA RESULTS ARE IDENTICAL TO THOSE EXPECTED IN THE MODE 4
                                ;TESTS.
                                ;
                                ;*****************************************************************************
                                ;TEST 162       TEST MODE 6 W/ DOP INSTS.
                                ;*****************************************************************************
011406                          TS162:
011406  012700  011332                  MOV     #TBL1-4,R0      ;INITIALIZE R0
011412  016037  000002  011336          MOV     2(R0),@#TBL1    ;TBL1=125252
011420  066037  000000  011336          ADD     0(R0),@#TBL1    ;TBL1=000377
011426  146037  177777  011336          BICB    -1(R0),@#TBL1   ;TBL1=000252
011434  156037  177776  011337          BISB    -2(R0),@#TBL1+1 ;TBL1=125252
011442  026037  177774  011336          CMP     -4(R0),@#TBL1   ;CHECK RESULT
011450  001401                          BEQ     TS163
011452  104000                          EMT                     ;RESULT OF MODE 6 INSTS. INCORRECT
                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 7 DOUBLE OPERAND INSTRUCTIONS.
                                ;THIS TEST USES THE SAME ADDRESS TABLE AND DATA TABLE USED BY
```

```
                                        ;THE MODE 5 TESTS.  THIS TIME THE DATA IS ACCESSED USING MODE 7.
                                        ;RO IS SET TO POINT TO THE MIDDLE OF THE ADDRESS TABLE IN THE MODE 5
                                        ;TEST.  THE TABLE IS ACCESSED FROM BOTTOM TO TOP BY VARYING THE OFFSET
                                        ;IN THE MODE 7 INSTRUCTIONS.  THE DATA RESULTS ARE IDENTICAL TO
                                        ;THOSE EXPECTED IN THE MODE 5 TESTS.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 163        TEST MODE 7 W/ DOP INSTS.
                                        ;*******************************************************************************
011454                                  TS163:
011454  012700  011400                          MOV     #TBL2-4,R0      ;INITIALIZE R0
011460  017037  000004  011336                  MOV     @4(R0),@#TBL1   ;TBL1=125252
011466  067037  000002  011336                  ADD     @2(R0),@#TBL1   ;TBL1=000377
011474  147037  000000  011336                  BICB    @0(R0),@#TBL1   ;TBL1=000252
011502  157037  177776  011337                  BISB    @-2(R0),@#TBL1+1                ;TBL1=125252
011510  027037  177774  011336                  CMP     @-4(R0),@#TBL1          ;CHECK RESULT
011516  001401                                  BEQ     TS164
011520  104000                                  EMT                     ;RESULT OF MODE 7 INSTS INCORRECT
                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES THE ROTATE MODE 0 INSTRUCTIONS.
                                        ;R0 IS LOADED WITH A DATA PATTERN, THE C-BIT IS LOADED, AND
                                        ;AN ROL INSTRUCTION IS EXECUTED WITH MODE 0.  THE OPERATION IS CHECKED
                                        ;BY TESTING THE RESULTING DATA AND THE STATE OF THE C AND V BITS.
                                        ;NEXT, THE SAME PROCEDURE IS EXECUTED TO TEST MODE 0 BYTE INSTRUCTIONS.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 164        TEST ROTATE INSTRUCTIONS OF MODE 0
                                        ;*******************************************************************************
011522                                  TS164:
011522  012700  125252                          MOV     #125252,R0      ;INITIALIZE DATA
011526  000261                                  SEC                     ;SET C-BIT
011530  006100                                  ROL     R0              ;TRY ROL W/ MODE 0
011532  102004                                  BVC     ROT0A           ;CC=0011
011534  103003                                  BCC     ROT0A
011536  022700  052525                          CMP     #052525,R0      ;CHECK DATA
011542  001401                                  BEQ     ROT0B
011544                                  ROT0A:
011544  104000                                  EMT                     ;ROL MODE 0 FAILED
011546  012700  125252                  ROT0B:  MOV     #125252,R0      ;INITIALIZE DATA
011552  000261                                  SEC                     ;SET C-BIT
011554  106100                                  ROLB    R0              ;TRY ROL W/ MODE 0 EVEN BYTE
011556  102004                                  BVC     ROT0C           ;CC=0011
011560  103003                                  BCC     ROT0C
011562  022700  125125                          CMP     #125125,R0      ;CHECK DATA
011566  001401                                  BEQ     TS165
011570                                  ROT0C:
011570  104000                                  EMT                     ;ROLB MODE 0 FAILED

                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;       THIS TEST VERIFIES THE ROTATE MODE 1 INSTRUCTIONS.
                                        ;THE DATA TO BE ROTATED IS IN LOC 0.  R0 IS USED AS THE
                                        ;ADDRESSING REGISTER.  THE C-BIT IS LOADED AND AN ROL IS EXECUTED.
                                        ;THE RESULTS ARE CHECKED BY COMPARING THE DATA RESULTS AND TESTING
                                        ;THE C AND V BITS.  THIS PROCEDURE IS THEN REPEATED TWICE MORE
```

```
                                        ;TO TEST THE BYTE ROTATES.  FIRST ON BYTE 0, THEN ON BYTE 1.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 165       TEST ROTATE INSTRUCTIONS W/ MODE 1
                                        ;*******************************************************************************
011572                                  TS165:
011572  005000                                  CLR     R0                      ;POINT TO LOC. 0
011574  012710  052525                          MOV     #52525,(R0)             ;INITIALIZE DATA
011600  000241                                  CLC                             ;CLEAR C-BIT
011602  006110                                  ROL     (R0)                    ;TRY ROL W/ MODE 1
011604  102005                                  BVC     ROT1A                   ;CC=1010
011606  103404                                  BCS     ROT1A
011610  023727  000000  125252                  CMP     @#0,#125252             ;CHECK RESULT
011616  001401                                  BEQ     ROT1B
011620                                  ROT1A:
011620  104000                                  EMT                             ;ROL MODE 1 FAILED
011622  000261                          ROT1B:  SEC
011624  012710  125252                          MOV     #125252,(R0)            ;INITIALIZE DATA
011630  106110                                  ROLB    (R0)                    ;TRY ROLB W/ MODE 1 EVEN BYTE
011632  102005                                  BVC     ROT1C                   ;CC=1011
011634  103004                                  BCC     ROT1C
011636  022737  125125  000000                  CMP     #125125,@#0             ;TEST RESULT
011644  001401                                  BEQ     ROT1D
011646                                  ROT1C:
011646  104000                                  EMT                             ;ROLB W/ MODE 1 EVEN BYTE FAILED
011650  012710  125252                  ROT1D:  MOV     #125252,(R0)
011654  005000                                  CLR     R0                      ;POINT TO ODD BYTE
011656  005200                                  INC     R0
011660  000261                                  SEC                             ;SET C-BIT
011662  106110                                  ROLB    (R0)                    ;TRY ROLB W/ MODE 1 ODD BYTE
011664  102005                                  BVC     ROT1E                   ;CC=0011
011666  103004                                  BCC     ROT1E
011670  022737  052652  000000                  CMP     #052652,@#0             ;CHECK DATA
011676  001401                                  BEQ     TS166
011700                                  ROT1E:
011700  104000                                  EMT                             ;ROLB W/ MODE 1 ODD BYTE FAILED

                                        ;*******************************************************************************
                                        ;
                                        ;               THIS TEST VERIFIES MODE 2 ROTATE INSTRUCTIONS.
                                        ;THE SAME PROCEDURE AS IN THE OTHER ROTATE TESTS ARE USED.  R0
                                        ;IS USED AS THE ADDRESSING REGISTER AND IS CHECKED FOR PROPER
                                        ;INCREMENTING.  BYTE INSTRUCTIONS ARE ALSO CHECKED.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 166       TEST ROTATE INSTRUCTIONS W/ MODE 2
                                        ;*******************************************************************************
011702                                  TS166:
011702  005000                                  CLR     R0                      ;POINT TO LOC 0
011704  012710  173737                          MOV     #173737,(R0)            ;INITIALIZE DATA
011710  000241                                  CLC                             ;CLEAR C-BIT
011712  006120                                  ROL     (R0)+                   ;TRY ROL W/ MODE 2
011714  103007                                  BCC     ROT2A                   ;CHECK C-BIT
011716  022737  167676  000000                  CMP     #167676,@#0             ;CHECK DATA
011724  001003                                  BNE     ROT2A                   ;BRANCH IF RESULT INCORRECT
011726  005300                                  DEC     R0                      ;TEST R0
```

```
011730  005300                          DEC     R0
011732  001401                          BEQ     ROT2B
011734                          ROT2A:
011734  104000                          EMT                             ;ROL W/ MODE 2 FAILED
011736  005000                  ROT2B:  CLR     R0                      ;POINT TO LOC 0
011740  012710  004040                  MOV     #4040,(R0)              ;INITIALIZE DATA
011744  000241                          CLC                             ;CLEAR C-BIT
011746  106120                          ROLB    (R0)+                   ;TRY ROLB W/ MODE 2 EVEN BYTE
011750  103406                          BCS     ROT2C                   ;CHECK C-BIT
011752  022737  004100  000000          CMP     #4100,@#0               ;CHECK DATA
011760  001002                          BNE     ROT2C                   ;BRANCH IF DATA INCORRECT
011762  005300                          DEC     R0                      ;CHECK R0
011764  001401                          BEQ     ROT2D
011766                          ROT2C:
011766  104000                          EMT                             ;ROLB W/ MODE 2 EVEN BYTE FAILED
011770  005000                  ROT2D:  CLR     R0                      ;POINT TO LOC 0
011772  012710  004040                  MOV     #4040,(R0)              ;INITIALIZE DATA
011776  005200                          INC     R0                      ;POINT TO ODD BYTE OF DATA
012000  000261                          SEC                             ;SET C-BIT
012002  106120                          ROLB    (R0)+                   ;TRY ROL W/ MODE 2 ODD BYTE
012004  103407                          BCS     ROT2E                   ;CHECK C-BIT
012006  022737  010440  000000          CMP     #10440,@#0              ;CHECK DATA
012014  001003                          BNE     ROT2E                   ;BRANCH IF DATA INCORRECT
012016  005300                          DEC     R0                      ;CHECK R0
012020  005300                          DEC     R0
012022  001401                          BEQ     TS167
012024                          ROT2E:
012024  104000                          EMT                             ;ROLB W/ MODE 2 ODD BYTE FAILED

                                ;*******************************************************************************
                                ;
                                ;               THIS TEST VERIFIES MODE 3 ROTATE INSTRUCTIONS.
                                ;THIS TEST USES THE SAME PROCEDURES AS IN THE OTHER ROTATE
                                ;TESTS.  THE DATA IS STORED IN LOC. 0 AND IS ADDRESSED USING
                                ;MODE 37.  BYTE ADDRESSING IS ALSO CHECKED FOR EVEN AND ODD BYTES.
                                ;
                                ;*******************************************************************************
                                ;TEST 167       TEST ROTATE INSTRUCTIONS /W MODE 3
                                ;*******************************************************************************
012026                          TS167:
012026  012737  052525  000000          MOV     #52525,@#0              ;INITIALIZE DATA IN LOC 0
012034  000261                          SEC                             ;SET C-BIT
012036  006137  000000                  ROL     @#0                     ;TRO ROL W/ MODE 3
012042  103404                          BCS     ROT3A                   ;CHECK C-BIT
012044  022737  125253  000000          CMP     #125253,@#0             ;CHECK DATA
012052  001401                          BEQ     ROT3B
012054                          ROT3A:
012054  104000                          EMT                             ;ROL W/ MODE 3 FAILED
012056  012737  125252  000000  ROT3B:  MOV     #125252,@#0             ;INITIALIZE DATA
012064  000241                          CLC                             ;CLEAR C-BIT
012066  106137  000000                  ROLB    @#0                     ;TRY ROL W/ MODE 3 EVEN BYTE
012072  103004                          BCC     ROT3C                   ;CHECK C-BIT
012074  023727  000000  125124  4$:     CMP     @#0,#125124             ;CHECK DATA
012102  001401                          BEQ     ROT3D
012104                          ROT3C:
012104  104000                          EMT                             ;ROL W/ MODE 3 EVEN BYTE FAILED
```

```
012106  012737  125252  000000  ROT3D:  MOV     #125252,@#0         ;INITIALIZE DATA IN LOC. 0
012114  000261                          SEC                         ;SET C-BIT
012116  106137  000001                  ROLB    @#1                 ;TRY ROL W/ MODE 3 ODD BYTE
012122  103004                          BCC     ROT3E               ;CHECK C-BIT
012124  022737  052652  000000          CMP     #052652,@#0         ;CHECK DATA
012132  001401                          BEQ     TS170
012134                          ROT3E:
012134  104000                          EMT                         ;ROL W/ MODE 3 ODD BYTE FAILED

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 4 ROTATE INSTRUCTIONS.  THE DATA IS
                                ;STORED IN LOC. 0.  R0 IS SET TO 2 AND THE CARRY IS SET.  AN ROL MODE 4
                                ;IS USED TO ROTATE LOCATION 0 USING R0.  THE DATA IS CHECKED
                                ;AND THE C AND V BITS ARE TESTED.  THE PROPER DECREMENTING OF
                                ;R0 IS VERIFIED.
                                ;
                                ;*******************************************************************************
                                ;TEST 170       TEST MODE 4 W/ ROTATE INSTRUCTIONS
                                ;*******************************************************************************
012136                          TS170:
012136  012737  070707  000000          MOV     #070707,@#0         ;INITIALIZE DATA IN LOC. 0
012144  012700  000002                  MOV     #2,R0               ;INITIALIZE R0 AS POINTER
012150  000261                          SEC                         ;SET C-BIT
012152  006140                          ROL     -(R0)               ;TRY ROL W/ MODE 4
012154  103406                          BCS     ROT4                ;CHECK C-BIT
012156  022737  161617  000000          CMP     #161617,@#0         ;CHECK DATA
012164  001002                          BNE     ROT4                ;BRANCH IF DATA INCORRECT
012166  005700                          TST     R0                  ;CHECK MODE 4 REGISTER
012170  001401                          BEQ     TS171
012172                          ROT4:
012172  104000                          EMT                         ;ROL MODE 4 FAILED

                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 5 ROTATE INSTRUCTIONS.
                                ;THE DATA IS STORED IN A WORK LOCATION (ROTX) AT THE END OF THE
                                ;TEST CODE.  LOC. 0 IS LOADED WITH THE ADDRESS OF THE DATA (ROTX).
                                ;R0 IS SET TO 2.  THE CARRY IS CLEARED AND A MODE 5 ROL
                                ;IS EXECUTED USING R0 AS AN ADDRESSING REGISTER.  THE DATA IS
                                ;CHECKED, THE C AND V BITS TESTED, AND R0 CHECKED FOR PROPER
                                ;DECREMENTING.
                                ;
                                ;*******************************************************************************
                                ;TEST 171       TEST MODE 5 W/ ROTATE INSTRUCTIONS
                                ;*******************************************************************************
012174                          TS171:
012174  012737  012240  000000          MOV     #ROTX,@#0           ;MOVE POINTER TO LOC. 0
012202  012700  000002                  MOV     #2,R0               ;SET MODE 5 REG. TO LOC. 0
012206  012767  107070  000024          MOV     #107070,ROTX        ;INITIALIZE DATA
012214  000241                          CLC                         ;CLEAR C-BIT
012216  006150                          ROL     @-(R0)              ;TRY ROL W/ MODE 5
012220  103006                          BCC     ROT5                ;CHECK C-BIT
012222  022737  016160  012240          CMP     #016160,@#ROTX      ;CHECK DATA
012230  001002                          BNE     ROT5                ;BRANCH IF DATA INCORRECT
012232  005700                          TST     R0                  ;CHECK MODE 5 REGISTER
```

```
012234  001402                          BEQ     TS172
012236                          ROT5:
012236  104000                          EMT                     ;ROL MODE 5 FAILED
012240  000000                  ROTX:   0

                                ;*************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 6 ROTATE INSTRUCTIONS.
                                ;IT USES THE SAME PROCEDURE AS THE ABOVE TEST EXCEPT THE
                                ;ROTATE INSTRUCTION USES MODE 6 ADDRESSING WITH REGISTER 7.
                                ;THE DATA IS STILL OPERATED ON IN LOC. ROTX (SEE PREVIOUS TEST).
                                ;
                                ;*************************************************************************
                                ;TEST 172       TEST MODE 6 W/ ROTATE INSTRUCTIONS
                                ;*************************************************************************
012242                          TS172:
012242  012737 125252 012240            MOV     #125252,@#ROTX  ;INITIALIZE DATA
012250  000261                          SEC                     ;SET C-BIT
012252  006167 177762                   ROL     ROTX            ;TRY ROL W/ MODE 6
012256  103004                          BCC     ROT6            ;CHECK C-BIT
012260  022737 052525 012240            CMP     #52525,@#ROTX   ;CHECK DATA
012266  001401                          BEQ     TS173
012270                          ROT6:
012270  104000                          EMT                     ;ROL W/ MODE 6 FAILED

                                ;*************************************************************************
                                ;
                                ;THIS TEST VERIFIES MODE 7 ROTATE INSTRUCTIONS.
```

```
                                       ;THE DATA IS SET IN LOC. ROTX, (SEE PREVIOUS TEST).  THE ROL INSTRUCTION
                                       ;ADDRESSES IT INDIRECTLY USING MODE 7 AND INDIRECT ADDRESS LOCATION
                                       ;(ROTXAD) FOLLOWING THE TEST CODE.
                                       ;
                                       ;*************************************************************************
                                       ;TEST 173        TEST MODE 7 W/ ROTATE INSTRUCTIONS
                                       ;*************************************************************************
012272                                 TS173:
012272  012737  052525  012240                 MOV     #52525,@#ROTX    ;INITIALIZE DATA
012300  012737  012240  012330                 MOV     #ROTX,@#ROTXAD   ;INITIALIZE ADDRESS POINTER
012306  000241                                 CLC                      ;CLEAR C-BIT
012310  006177  000014                         ROL     @ROTXAD          ;TRY ROL W/ MODE 7
012314  103404                                 BCS     ROT7             ;CHECK C-BIT
012316  023727  012240  125252                 CMP     @#ROTX,#125252   ;CHECK DATA
012324  001402                                 BEQ     TS174
012326                                 ROT7:
012326  104000                                 EMT                      ;ROL W/ MODE 7 FAILED
012330  000000                         ROTXAD: 0


                                       ;*************************************************************************
                                       ;
                                       ;        THIS TEST VERIFIES MODE 0 SWAB INSTRUCTION.  R0 IS SET TO
                                       ;177400.  A SWAB MODE 0 IS EXECUTED AND THE CONDITIONAL BRANCH
                                       ;IS USED TO CHECK THE SIGN OF THE RESULT.  ALSO, A COMPARISON
                                       ;IS MADE TO CHECK THE DATA RESULTS.
                                       ;
                                       ;*************************************************************************
                                       ;TEST 174        TEST MODE 0 W/ SWAB INST.
                                       ;*************************************************************************
012332                                 TS174:
012332  012700  177400                         MOV     #177400,R0       ;MOVE TEST PATTERN TO R0
012336  000300                                 SWAB    R0               ;TRY SWAB MODE 0
012340  100401                                 BMI     SB0
012342  104000                                 EMT                      ;SWAB DID NOT SET CC'S CORRECT
012344  022700  000377                 SB0:    CMP     #377,R0          ;CHECK RESULT
012350  001401                                 BEQ     TS175
012352  104000                                 EMT                      ;RESULT OF SWAB MODE 0 FAILED

                                       ;*************************************************************************
                                       ;
                                       ;        THIS TEST VERIFIES MODE 1 SWAB INSTRUCTION.  THE TEST
                                       ;PATTERN IS MOVED TO LOC 0.  R0 IS CLEARED AND USED AS THE ADDRESSING
                                       ;REGISTER IN THE MODE 1 SWAB.  THE DATA RESULTS ARE CHECKED WITH
                                       ;A COMPARE.
                                       ;
                                       ;*************************************************************************
                                       ;TEST 175        TEST MODE 1 W/ SWAB INST
                                       ;*************************************************************************
012354                                 TS175:
012354  012737  125652  000000                 MOV     #125652,@#0      ;MOVE TEST PATTERN TO LOC. 0
012362  005000                                 CLR     R0               ;R0=0
012364  000310                                 SWAB    (R0)             ;TRY SWAB MODE 1
012366  022737  125253  000000                 CMP     #125253,@#0      ;CHECK RESULT
012374  001401                                 BEQ     TS176
012376  104000                                 EMT                      ;RESULT OF SWAB MODE 1 FAILED
```

```


                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES MODE 2 SWAB INSTRUCTION.  THE TEST
                                        ;PATTERN IS MOVED TO LOC 0.  R0 IS CLEARED AND USED AS THE MODE
                                        ;2 ADDRESSING REGISTER.  THE RESULTS ARE CHECKED WITH A COMPARE.
                                        ;R0 IS CHECKED FOR PROPER DECREMENTING.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 176       TEST MODE 2 W/ SWAB INST
                                        ;*******************************************************************************
 012400                                 TS176:
 012400  012737  125152  000000                 MOV     #125152,@#0     ;MOVE TEST PATTERN TO LOC. 0
 012406  005000                                 CLR     R0              ;R0=0
 012410  000320                                 SWAB    (R0)+           ;TRY SWAB MODE 2
 012412  022737  065252  000000                 CMP     #65252,@#0      ;CHECK RESULT
 012420  001401                                 BEQ     SB2
 012422  104000                                 EMT                     ;RESULT OF SWAB MODE 0 FAILED
 012424  162700  000002                 SB2:    SUB     #2,R0           ;CHECK EFFECT OF REG.
 012430  001401                                 BEQ     TS177
 012432  104000                                 EMT                     ;REGISTER VALUE INCORRECT


                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES MODE 3 SWAB INSTRUCTION.  THE TEST
                                        ;PATTERN IS MOVED TO LOC 0.  A MODE 3 SWAB INSTRUCTION IS EXECUTED
                                        ;USING R7 AS THE ADDRESSING REGISTER.  A COMPARE VERIFIES THE
                                        ;DATA RESULTS.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 177       TEST MODE 3 W/SWAB INST.
                                        ;*******************************************************************************
 012434                                 TS177:
 012434  012737  000377  000000                 MOV     #377,@#0        ;MOVE TEST PATTERN TO LOC. 0
 012442  000337  000000                         SWAB    @#0             ;TRY SWAB W/ MODE 3
 012446  022737  177400  000000                 CMP     #177400,@#0     ;CHECK RESULT
 012454  001401                                 BEQ     TS200
 012456  104000                                 EMT                     ;RESULT OF SWAB INCORRECT


                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES MODE 4 SWAB INSTRUCTIONS.  THE DATA
                                        ;IS MOVED TO LOC 0.  R0 IS SET TO 2 AND USED AS THE MODE 4 ADDRESSING
                                        ;REGISTER.  THE DATA IS CHECKED WITH A COMPARE AND R0 IS CHECKED
                                        ;FOR PROPER DECREMENTING.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 200       TEST MODE 4 W/ SWAB INST
                                        ;*******************************************************************************
 012460                                 TS200:
 012460  012737  125652  000000                 MOV     #125652,@#0     ;MOVE TEST PATTERN TO LOC. 0
 012466  012700  000002                         MOV     #2,R0           ;SET UP REGISTER POINTER
 012472  000340                                 SWAB    -(R0)           ;TRY SWAB MODE 4
```

```
012474  022737  125253  000000          CMP     #125253,@#0       ;CHECK RESULT
012502  001401                          BEQ     SB4
012504  104000                          EMT                       ;RESULT OF SWAB INCORRECT
012506  005700                  SB4:    TST     R0                ;CHECK EFFECT ON REG.
012510  001401                          BEQ     TS201
012512  104000                          EMT                       ;REGISTER VALUE INCORRECT


                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 5 SWAB INSTRUCTION.  THE TEST USES
                                ;TWO LOCATIONS FOLLOWING THE TEST CODE.  SB5X HOLDS THE DATA;
                                ;SB5XAD IS A POINTER TO THE DATA LOCATION.  THE DATA IS MOVED TO
                                ;SB5X AND R0 IS SET TO TWO PLUS THE ADDRESS OF SB5XAD.  FOLLOWING
                                ;THE MODE 5 SWAB SB5X IS CHECKED FOR THE PROPER DATA.  R0 IS
                                ;CHECKED TO SEE THAT IT WAS DECREMENTED PROPERLY.
                                ;
                                ;*****************************************************************************
                                ;TEST 201        TEST MODE 5 W/ SWAB INST.
                                ;*****************************************************************************
012514                          TS201:
012514  012700  012556                  MOV     #SB5XAD+2,R0      ;SET UP POINTER TO WORK LOCATION
012520  012767  125125  000024          MOV     #125125,SB5X      ;MOVE PATTERN TO WORK LOCATION
012526  000350                          SWAB    @-(R0)            ;TRY SWAB MODE 5
012530  022767  052652  000014          CMP     #52652,SB5X       ;CHECK RESULT
012536  001401                          BEQ     SB5A
012540  104000                          EMT                       ;RESULT OF SWAB INCORRECT
012542  020027  012554          SB5A:   CMP     R0,#SB5XAD        ;CHECK RESULT OF REG.
012546  001403                          BEQ     TS202
012550                          SB5:
012550  104000                          EMT                       ;REGISTER VALUE INCORRECT
012552  000000                  SB5X:   0                         ;WORK LOCATION
012554  012552                  SB5XAD: SB5X


                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 6 SWAB INSTRUCTION.  THIS TEST
                                ;USES A WORK LOCATION (SB6X) FOLLOWING THE TEST CODE.  TEST DATA
                                ;IS LOADED INTO THE WORK LOCATION.  R0, THE ADDRESSING REGISTER
                                ;IS LOADED WITH 6 LESS THEN THE ADDRESS OF THE WORK LOCATION.
                                ;THE MODE 6 SWAB IS EXECUTED WITH A +6 OFFSET.  THE DATA IS
                                ;VERIFIED WITH A COMPARE.
                                ;
                                ;*****************************************************************************
                                ;TEST 202        TEST MODE 6 W/ SWAB INST.
                                ;*****************************************************************************
012556                          TS202:
012556  012767  125125  000022          MOV     #125125,SB6X      ;MOVE PATTERN TO WORK LOCATION
012564  012700  012600                  MOV     #SB6X-6,R0        ;MOVE OFFSET POINTER TO R0
012570  000360  000006                  SWAB    6(R0)             ;TRY SWAB W/ MODE 6
012574  022760  052652  000006          CMP     #52652,6(R0)      ;CHECK RESULT
012602  001402                          BEQ     TS203
012604                          SB6:
012604  104000                          EMT                       ;RESULT OF SWAB INCORRECT
012606  000000                  SB6X:   0                         ;WORK LOCATION
```

```


                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES MODE 7 SWAB INSTRUCTION.  THIS TEST
                                        ;USES TWO LOCATIONS FOLLOWING THE TEST CODE:  A WORK LOCATION
                                        ;(SB7X) AND A POINTER TO THE WORK LOCATION (SB7XAD).  DATA IS MOVED
                                        ;TO THE WORK LOCATION.  R0 IS LOADED WITH 72 LESS THAN THE ADDRESS
                                        ;OF THE ADDRESS POINTER.  THE DATA IS SWAB'ED USING A MODE 7
                                        ;INSTRUCTION WITH AN OFFSET OF +72.  THE DATA IS VERIFIED WITH A
                                        ;COMPARE.
                                        ;
                                        ;*******************************************************************************
                                        ;TEST 203       TEST MODE 7 W/ SWAB INST.
                                        ;*******************************************************************************
 012610                                 TS203:
 012610 012767  177400  000022                  MOV     #177400,SB7X    ;MOVE PATTERN TO WORK LOCATION
 012616 012700  012550                          MOV     #SB7XAD-72,R0   ;MOVE OFFSET POINTER TO R0
 012622 000370  000072                          SWAB    @72(R0)         ;TRY SWAB MODE 7
 012626 027027  000072  000377                  CMP     @72(R0),#377    ;CHECK RESULTS
 012634 001403                                  BEQ     TS204
 012636                                 SB7:
 012636 104000                                  EMT                     ;RESULT OF SWAB INCORRECT
 012640 000000                          SB7X:   0                       ;WORK LOCATION
 012642 012640                          SB7XAD: SB7X                    ;POINTER TO WORK LOCATION


                                        ;*******************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES ALL LEGAL MODES OF THE JMP INSTRUCTION.
                                        ;BECAUSE OF THE NATURE OF THE INSTRUCTION UNDER TEST, THIS TEST
                                        ;UTILIZES SEVERAL DIFFERENT TECHNIQUES.  THE CODE IS NOT EXECUTED
                                        ;IN A LINEAR FASHION.  THE DIFFERENT MODES ARE EXECUTED IN ORDER
                                        ;FROM 1-7; HOWEVER, THE CODE IS ARRANGED SO THAT CONTROL LEAP
                                        ;FROGS THRU THE TEST CODE.  THE ORDER OF APPEARANCE OF THE CODE
                                        ;IS:
                                        ;       JMP MODE 1
                                        ;       JMP MODE 3
                                        ;       JMP MODE 2
                                        ;       JMP MODE 4
                                        ;       JMP MODE 6
                                        ;       JMP MODE 5
                                        ;       JMP MODE 7
                                        ;AN INTERNAL SEQUENCE TEST (JMPSEQ) IS USED TO INSURE THAT THE
                                        ;JUMPS ARE OCCURRING IN THE PROGRAMMED SEQUENCE.
                                        ;       THE TEST IS MADE UP OF SEVERAL BLOCKS OF CODE.  EACH CODE
                                        ;BEGINS WITH A LABEL WHICH INDICATES THE MODE BEING EXECUTED IN
                                        ;THAT BLOCK.  A SIMPLE PROCEDURE IS FOLLOWED IN EACH BLOCK.  FOR
                                        ;EXAMPLE THE CODE BEGINNING AT JMP3 WILL FIRST COMPARE THE RESULTS
                                        ;OF THE PREVIOUS MODE 2 JUMP.  (ANY REGISTER CHANGES ARE VERIFIED
                                        ;AND THE SEQUENCE CHECK IS MADE).  THEN THE REGISTERS ARE SETUP
                                        ;FOR A MODE 3 JUMP TO THE NEXT TEST BLOCK (HERE, JMP4), THE SEQUENCE
                                        ;CHECKER IS UPDATED AND THE JUMP IS EXECUTED.
                                        ;       IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN
                                        ;DETERMINING JUST WHICH MODE FAILED.  IF THE SEQUENCE IS CORRECT
                                        ;THEN THE ERROR DETECTED WAS A MODE FAILURE (E.G. FAILURE OF THE
```

```
                                        ;REGISTER TO BE INCREMENTED IN MODE 2 JUMP.)
                                        ;
                                        ;*************************************************************************
                                        ;TEST 204        TEST THE JMP INSTRUCTION IN ALL MODES
                                        ;*************************************************************************
 012644                                 TS204:
 012644  005067  000240                         CLR     JMPSEQ          ;ESTABLISH A SEQUENCE CHECKER
 012650  012700  012714                         MOV     #JMP2,R0        ;SET R0=JUMP TARGET
 012654  000110                                 JMP     (R0)            ;TRY JMP MODE 1
 012656  022700  012660                 JMP3:   CMP     #.+2,R0         ;CHECK RESULT OF MODE 2 JUMP
 012662  001401                                 BEQ     JMP3A
 012664  104000                                 EMT                     ;REGISTER VALUE AFTER JMP MODE 2 INCORRECT
 012666  026727  000216  000001         JMP3A:  CMP     JMPSEQ,#1       ;MAKE SURE JMPS ARE IN SEQUENCE: JMPSEQ=1?
 012674  001401                                 BEQ     JMP3B
 012676  104000                                 EMT                     ;SHOULD BE HERE FROM JMP MODE 2 ONLY
 012700  012700  012712                 JMP3B:  MOV     #IJMP4,R0       ;POINT R0 TO INDIRECT JMP ADDR.
 012704  005267  000200                         INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
 012710  000130                                 JMP     @(R0)+          ;TRY JMP MODE 3
 012712  012736                         IJMP4:  JMP4                    ;ADDRESS INDIRECT JUMP

 012714  005767  000170                 JMP2:   TST     JMPSEQ          ;CHECK THAT JMPS ARE IN SEQUENCE: JMPSEQ=0?
 012720  001401                                 BEQ     JMP2A
 012722  104000                                 EMT                     ;SHOULD BE HERE FROM JMP MODE 1 ONLY
 012724  005267  000160                 JMP2A:  INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
 012730  012700  012656                         MOV     #JMP3,R0        ;SET R0=JUMP TARGET
 012734  000120                                 JMP     (R0)+           ;TRY A JUMP MODE 2 TO "JMP3"
 012736  022700  012714                 JMP4:   CMP     #IJMP4+2,R0     ;CHECK RESULT OF REGISTER IN MODE 3 JUMP
 012742  001401                                 BEQ     JMP4A
 012744  104000                                 EMT                     ;REGISTER VALUE AFTER MODE 3 JUMP INCORRECT
 012746  022767  000002  000134         JMP4A:  CMP     #2,JMPSEQ       ;CHECK JUMP SEQUENCE: JMPSEQ=2?
 012754  001401                                 BEQ     JMP4B
 012756  104000                                 EMT                     ;SHOULD BE ONLY FROM MODE 3 JUMP
 012760  012700  013022                 JMP4B:  MOV     #JMP5+2,R0      ;SET UP POINTER TO JUMP TARGET
 012764  005267  000120                         INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
 012770  000140                                 JMP     -(R0)           ;TRY JUMP MODE 4 TO "JMP4"

 012772  022767  000004  000110         JMP6:   CMP     #4,JMPSEQ       ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=4?
 013000  001401                                 BEQ     JMP6A
 013002  104000                                 EMT                     ;SHOULD BE HERE ONLY FROM MODE 5 JUMP
 013004  012700  013444                 JMP6A:  MOV     #JMP7+376,R0    ;SET UP OFFSET POINTER TO JUMP TARGET
 013010  005267  000074                         INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
 013014  000160  177402                         JMP     -376(R0)        ;TRY MODE 6 JUMP

 013020  022767  000003  000062         JMP5:   CMP     #3,JMPSEQ       ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=3?
 013026  001401                                 BEQ     JMP5A
 013030  104000                                 EMT                     ;SHOULD ONLY BE HERE FROM MODE 4 JUMP
 013032  012700  013046                 JMP5A:  MOV     #IJMP5+2,R0     ;SET UP POINTER TO INDIRECT JUMP ADDR.
 013036  005267  000046                         INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
 013042  000150                                 JMP     @-(R0)          ;TRY JUMP MODE 5 TO "JMP6"
 013044  012772                         IJMP5:  JMP6                    ;INDIRECT ADDRESS POINTER

 013046  022767  000005  000034         JMP7:   CMP     #5,JMPSEQ       ;CHECK JUMPS IN SEQUENCE: JMPSEQ=5?
 013054  001401                                 BEQ     JMP7A
 013056  104000                                 EMT                     ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
 013060  012700  013104                 JMP7A:  MOV     #IJMP+10,R0     ;SET UP OFFSET POINTER TO INDIRECT ADDR.
 013064  005267  000020                         INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
```

```
013070  000170  177770                     JMP     @-10(R0)         ;TRY MODE 7 JUMP
013074  013076                   IJMP:     JMPCK                    ;INDIRECT ADDRESS

013076  026727  000006  000006   JMPCK:    CMP     JMPSEQ,#6        ;CHECK JUMPS IN SEQUENCE: JMPSEQ
013104  001402                             BEQ     TS205
013106  104000                             EMT                      ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
013110  000000                   JMPSEQ:   0

                                 ;*******************************************************************************
                                 ;
                                 ;          THIS TEST VERIFIES ALL LEGAL MODES OF THE JSR INSTRUCTION.
                                 ;THE CONCEPT OF LEAP FROGGING AND SEQUENCE CHECKING (JSRSEQ) IS
                                 ;IDENTICAL TO THAT USED IN JMP TEST (SEE PREVIOUS TEST).  EACH
                                 ;BLOCK OF CODE VERIFIES THE PREVIOUS JSR BY CHECKING THE SEQUENCE,
                                 ;CHECKING THAT THE PC WAS SAVED IN THE SPECIFIED REGISTER, CHECKING
                                 ;THAT THE SP WAS DECREMENTED, CHECKING THAT THE REGISTER WAS
                                 ;SAVED ON THE STACK, AND FINALLY CHECKING THAT ANY MODE ADDRESS
                                 ;REGISTER ALTERATIONS (E.G. INCREMENT REGISTER IN MODE 2) WERE
                                 ;SUCCESSFUL.  R1 IS USED AS THE REGISTER IN ALL JSR INSTRUCTIONS.
                                 ;          IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN
                                 ;DETERMINING JUST WHICH MODE FAILED.  IF THE SEQUENCE IS CORRECT
                                 ;THEN THE ERROR DETECTED WAS A FUNCTIONAL FAILURE (E.G., INCORRECT
                                 ;REGISTER SAVED).
                                 ;
                                 ;*******************************************************************************
                                 ;TEST 205          TEST JSR INSTRUCTION W/ ALL MODES
                                 ;*******************************************************************************
013112                           TS205:
013112  000402                             BR      JSR1
013114  000137  013476           JSR0:     JMP     @#JSRCK1

013120  012706  001000           JSR1:     MOV     #STBOT,R6        ;SET STACK POINTER
013124  012700  013216                     MOV     #JSR2,R0         ;SET TARGET ADDRESS
013130  005037  013456                     CLR     @#JSRSEQ         ;INITIALIZE SEQUENCE CHECKER
013134  005001                             CLR     R1               ;INITIALIZE R1
013136  005101                             COM     R1
013140  004110                             JSR     R1,(R0)          ;TRY JSR MODE 1
                                                                    ; TO SCOPE: REPLACE THE MOVE INSTRUCTION   <====
                                                                    ;           FOLLOWING W/ 774               <====
013142                           JSR1A:
013142  104000                             EMT                      ;JSR MODE 1 FAILED

013144  022737  000001  013456   JSR3:     CMP     #1,@# JSRSEQ     ;CHECK SEQUENCE: JSRSEQ=1?
013152  001014                             BNE     JSR3A            ;BRANCH IF OUT OF SEQUENCE
013154  020127  013272                     CMP     R1,#JSR4         ;PROPER PC SAVED?
013160  001011                             BNE     JSR3A            ;BRANCH IF PC WRONG
013162  022706  000776                     CMP     #STBOT-2,R6      ;STACK POINTER DECREMENTED?
013166  001006                             BNE     JSR3A            ;BRANCH IF SP WRONG
013170  022716  125252                     CMP     #125252,(R6)     ;REG SAVED ON STACK?
013174  001003                             BNE     JSR3A            ;BRANCH IF REG. NOT SAVED
013176  022700  013146                     CMP     #JSR3+2,R0       ;MODE 2 INCREMENT CORRECT?
013202  001401                             BEQ     JSR3B
013204                           JSR3A:
013204  104000                             EMT                      ;JSR MODE 3 MALFUNCTIONED
013206  005237  013456           JSR3B:    INC     @#JSRSEQ         ;UPDATE SEQUENCE CHECKER
013212  004137  013272                     JSR     R1,@#JSR4        ;TRY JSR MODE 4
```

```
013216 005737 013456         JSR2:   TST     @#JSRSEQ        ;CHECK SEQUENCE: JSRSEQ=0?
013222 001011                        BNE     JSR2A           ;BRANCH IF OUT OF SEQUENCE
013224 020127 013142                 CMP     R1,#JSR1A       ;PROPER PC SAVED?
013230 001006                        BNE     JSR2A           ;BRANCH IF PC WRONG
013232 022706 000776                 CMP     #STBOT-2,R6     ;R6 DECREMENT?
013236 001003                        BNE     JSR2A           ;BRANCH IF R6 IS INCORRECT
013240 021627 177777                 CMP     (R6),#-1        ;REGISTER SAVED?
013244 001401                        BEQ     JSR2B
013246                       JSR2A:
013246 104000                        EMT                     ;JSR MODE 1 MALFUNCTIONED
013250 012706 001000         JSR2B:  MOV     #STBOT,R6       ;INITIALIZE R6
013254 012701 125252                 MOV     #125252,R1      ;INITIALIZE R1
013260 005237 013456                 INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
013264 012700 013144                 MOV     #JSR3,R0        ;SET TARGET ADDRESS
013270 004120                        JSR     R1,(R0)+        ;TRY JSR MODE 2

013272 022737 000002 013456  JSR4:   CMP     #2,@#JSRSEQ     ;CHECK SEQUENCE: JSRSEQ=2?
013300 001003                        BNE     JSR4A           ;BRANCH IF OUT OF SEQUENCE
013302 022701 013216                 CMP     #JSR2,R1        ;PROPER PC SAVED?
013306 001401                        BEQ     JSR4B
013310                       JSR4A:
013310 104000                        EMT                     ;JSR MODE 3 MALFUNCTIONED
013312 005237 013456         JSR4B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
013316 012700 013364                 MOV     #JSR5+2,R0      ;SET TARGET ADDRESS
013322 004140                        JSR     R1,-(R0)        ;TRY JSR MODE 4

013324 022767 000004 000124  JSR6:   CMP     #4,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=4?
013332 001006                        BNE     JSR6A           ;BRANCH IF OUT OF SEQUENCE
013334 022701 013422                 CMP     #JSR7,R1        ;PROPER PC SAVED?
013340 001003                        BNE     JSR6A           ;BRANCH IF PC WRONG
013342 022700 013452                 CMP     #JSR6AD,R0      ;MODE 5 REGISTER CORRECT?
013346 001401                        BEQ     JSR6B
013350                       JSR6A:
013350 104000                        EMT                     ;JSR MODE 5 FAILED
013352 005237 013456         JSR6B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
013356 004167 000040                 JSR     R1,JSR7         ;TRY JSR MODE 6
013362 022767 000003 000066  JSR5:   CMP     #3,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=3?
013370 001006                        BNE     JSR5A           ;BRANCH IF OUT OF SEQUENCE
013372 022701 013324                 CMP     #JSR6,R1        ;PROPER PC SAVED?
013376 001003                        BNE     JSR5A           ;BRANCH IF PC WRONG
013400 022700 013362                 CMP     #JSR5,R0        ;CHECK MODE 4 REGISTER
013404 001401                        BEQ     JSR5B
013406                       JSR5A:
013406 104000                        EMT                     ;JSR MODE 4 MALFUNCTIONED
013410 005237 013456         JSR5B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
013414 012700 013454                 MOV     #JSR6AD+2,R0    ;POINT R0 TO TARGET ADDRESS
013420 004150                        JSR     R1,@-(R0)       ;TRY JSR MODE 5

013422 022737 000005 013456  JSR7:   CMP     #5,@#JSRSEQ     ;CHECK SEQUENCE: JSRSEQ=5?
013430 001003                        BNE     JSR7A           ;BRANCH IF OUT OF SEQUENCE
013432 022701 013362                 CMP     #JSR5,R1        ;PROPER PC SAVED?
013436 001401                        BEQ     JSR7B
013440                       JSR7A:
013440 104000                        EMT                     ;JSR MODE 6 FAILED
013442 005237 013456         JSR7B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
```

```
013446  004177  000002                  JSR     R1,@JSRCKAD     ;TRY JSR MODE 7
013452  013324                  JSR6AD: JSR6                    ;MODE 5 TARGET ADDRESS
013454  013460                  JSRCKAD:JSRCK                   ;MODE 7 TARGET ADDRESS
013456  000000                  JSRSEQ: 0                       ;SEQUENCE CHECKER
013460  022767  000006  177770  JSRCK:  CMP     #6,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=6?
013466  001003                          BNE     JSRCK1          ;BRANCH IF OUT OF SEQUENCE
013470  022701  013452                  CMP     #JSR6AD,R1      ;PROPER PC SAVED?
013474  001401                          BEQ     TS206
013476                          JSRCK1:
013476  104000                          EMT                     ;JSR MODE 7 MALFUNCTIONED
                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES THE RTS INSTRUCTION.  THE STACK POINTER
                                ;IS INITIALIZED AND A TEST PATTERN STORED ON STACK.  R0 IS LOADED
                                ;WITH RETURN ADDRESS.  AN RTS IS EXECUTED, AND, AT THE TARGET
                                ;ADDRESS, A CHECK IS MADE THAT R0 WAS PROPERLY RESTORED FROM THE
                                ;STACK.
                                ;*******************************************************************************
                                ;TEST 206       TEST RTS INSTRUCTION
                                ;*******************************************************************************
013500                          TS206:
013500  012706  001000                  MOV     #STBOT,R6       ;INITIALIZE STACK POINTER
013504  012746  052525                  MOV     #52525,-(R6)    ;INITIALIZE TOP OF STACK
013510  012700  013520                  MOV     #RTS1,R0        ;INITIALIZE RETURN REGISTER
013514  000200                          RTS     R0              ;TRY RTS THROUGH R0
                                                                ; TO SCOPE: REPLACE THE MOVE INSTRUCTION   <====
                                                                ;           FOLLOWING W/ 770               <====
013516  104000                          EMT                     ;RTS FAILED
013520  022700  052525          RTS1:   CMP     #52525,R0       ;CHECK THAT R0 RESTORED FROM STACK
013524  001401                          BEQ     TS207
013526  104000                          EMT                     ;RTS MALFUNCTIONED
                                ;*******************************************************************************
                                ;
                                ;       THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF A GROUP
                                ;OF FOUR INSTRUCTIONS.  THE GROUP CONSISTS OF THE INSTRUCTIONS:
                                ;MOV, BIC, BIT, AND BIS.  THESE INSTRUCTIONS ARE SIMILAR IN THE
                                ;WAY THEY EFFECT THE C AND V BITS.  THEY ALL LEAVE THE V-BIT
                                ;CLEAR AND THE C-BIT UNAFFECTED.
                                ;       THE TEST PROCEDURE IS AS FOLLOWS:  THE N, Z, AND V BITS
                                ;ARE LOADED WITH THE COMPLEMENT OF THE EXPECTED RESULTS, THE C-BIT
                                ;IS LOADED WITH THE DESIRED RESULT.  THE INSTRUCTION IS EXECUTED
                                ;WITH DIFFERENT DATA PATTERNS AND THE RESULTS ARE VERIFIED WITH
                                ;A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS.  THE DATA IS CHOSEN
                                ;TO PRODUCT ALL POSSIBLE COMBINATIONS OF THE C AND V BITS.
                                ;
                                ;*******************************************************************************
                                ;TEST 207       TEST MOV INSTRUCTION
                                ;*******************************************************************************
013530                          TS207:
013530  000277                          SCC                     ;CC=0110
013532  000251                          +CLN!CLC
013534  012700  100000                  MOV     #100000,R0      ;CC=1000
```

```
013540  101402                  BLOS    MOV1
013542  102401                  BVS     MOV1
013544  100401                  BMI     MOV2
013546                  MOV1:
013546  104000                  EMT                     ;MOV DID NOT SET CC'S CORRECTLY

013550  000277          MOV2:   SCC                     ;CC=1011
013552  000244                  CLZ
013554  012700  000000          MOV     #0,R0           ;CC=0101
013560  101002                  BHI     MOV3            ;C OR Z = 0?
013562  102401                  BVS     MOV3            ;V=1?
013564  100001                  BPL     TS210
013566                  MOV3:
013566  104000                  EMT                     ;MOV DID NOT SET CC'S CORRECTLY
                        ;*******************************************************************************
                        ;TEST 210       TEST BIT INSTRUCTION
                        ;*******************************************************************************
013570                  TS210:
013570  012700  100001          MOV     #100001,R0
013574  000277                  SCC                     ;CC=0110
013576  000251                  +CLN!CLC
013600  032700  100000          BIT     #100000,R0      ;CC=1000
013604  101402                  BLOS    BITST1
013606  102401                  BVS     BITST1
013610  100401                  BMI     BITST2
013612                  BITST1:
013612  104000                  EMT                     ;BIT DID NOT SET CC'S CORRECTLY

013614  000277          BITST2: SCC                     ;CC=1011
013616  000244                  CLZ
013620  032700  077776          BIT     #77776,R0       ;CC=0101
013624  101002                  BHI     BITST3
013626  102401                  BVS     BITST3
013630  100001                  BPL     TS211
013632                  BITST3:
013632  104000                  EMT                     ;BIT DID NOT SET CC'S CORRECTLY
                        ;*******************************************************************************
                        ;TEST 211       TEST BIC INSTRUCTION
                        ;*******************************************************************************
013634                  TS211:
013634  012700  177777          MOV     #177777,R0
013640  000277                  SCC                     ;CC=0110
013642  000251                  +CLN!CLC
013644  042700  077777          BIC     #77777,R0       ;CC=1000
013650  101402                  BLOS    BIC1
013652  102401                  BVS     BIC1
013654  100401                  BMI     BIC2
013656                  BIC1:
013656  104000                  EMT                     ;BIC DID NOT SET CC'S CORRECTLY
013660  000277          BIC2:   SCC                     ;CC=1011
013662  000244                  CLZ
013664  042700  100000          BIC     #100000,R0      ;CC=0101
013670  101002                  BHI     BIC3
013672  102401                  BVS     BIC3
013674  100001                  BPL     TS212
013676                  BIC3:
```

```
013676  104000                  EMT                             ;BIC DID NOT SET CC'S CORRECTLY
                        ;*******************************************************************************
                        ;TEST 212       TEST BIS INSTRUCTION
                        ;*******************************************************************************
013700                  TS212:
013700  005000                  CLR     R0                      ;R0=0
013702  000277                  SCC                             ;CC=1010
013704  000251                  +CLN!CLC
013706  052700  000000          BIS     #0,R0                   ;CC=0100  R0=0
013712  103403                  BCS     BIS1
013714  102402                  BVS     BIS1
013716  100401                  BMI     BIS1
013720  001401                  BEQ     BIS2
013722                  BIS1:
013722  104000                  EMT                             ;BIS DID NOT SET CC'S CORRECTLY
013724  000277          BIS2:   SCC                             ;CC=0111
013726  000250                  CLN
013730  052700  177777          BIS     #177777,R0              ;CC=1001
013734  103003                  BCC     BIS3
013736  102402                  BVS     BIS3
013740  001401                  BEQ     BIS3
013742  100401                  BMI     TS213
013744                  BIS3:
013744  104000                  EMT                             ;BIS DID NOT SET CC'S CORRECTLY

                        ;*******************************************************************************
                        ;
                        ;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE INC AND
                        ;DEC INSTRUCTIONS.  THESE INSTRUCTIONS BOTH EFFECT THE C AND V
                        ;BITS THE SAME; THE C-BIT IS LEFT UNCHANGED AND THE V-BIT IS DEPENDENT
                        ;UPON THE DATA RESULTS.  THE SAME PROCEDURE IS USED.  THE CONDITION
                        ;CODE BITS ARE INITIALIZED, THE INSTRUCTION IS EXECUTED AND THE
                        ;RESULTS ARE VERIFIED WITH A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS.
                        ;THIS PROCEDURE IS REPEATED WITH SEVERAL DATA PATTERNS TO PRODUCE
                        ;DIFFERENT COMBINATIONS OF THE C AND V BITS.
                        ;
                        ;*******************************************************************************
                        ;TEST 213       TEST INC INSTRUCTION
                        ;*******************************************************************************
013746                  TS213:
013746  012700  077777          MOV     #077777,R0              ;R0=077777
013752  000257                  CCC                             ;CC=0100
013754  000264                  SEZ
013756  005200                  INC     R0                      ;CC=1010   R0=10000
013760  101402                  BLOS    INC1
013762  100001                  BPL     INC1
013764  102401                  BVS     INC2
013766                  INC1:
013766  104000                  EMT                             ;INC DID NOT SET CC'S CORRECTLY
013770  052700  077777  INC2:   BIS     #77777,R0               ;R0=177777
013774  000261                  SEC                             ;CC=1011
013776  000244                  CLZ
014000  005200                  INC     R0                      ;CC=0101   R0=0
014002  100403                  BMI     INC3
014004  102402                  BVS     INC3
014006  103001                  BCC     INC3
```

```
014010  001401                          BEQ     INC4
014012                          INC3:
014012  104000                          EMT                             ;INC DID NOT SET CC'S CORRECTLY

014014  000277                  INC4:   SCC                             ;CC=1110
014016  000241                          CLC
014020  005200                          INC     R0                      ;CC=0000    R0=1
014022  101402                          BLOS    INC5
014024  100401                          BMI     INC5
014026  100001                          BPL     TS214
014030                          INC5:
014030  104000                          EMT                             ;INC DID NOT SET CC'S CORRECTLY

                                ;*****************************************************************************
                                ;TEST 214       TEST DEC INSTRUCTION
                                ;*****************************************************************************
014032                          TS214:
014032  012700  000002                  MOV     #2,R0                   ;R0=2
014036  000277                          SCC                             ;CC=1111
014040  005300                          DEC     R0                      ;CC=0001    R0=1
014042  100403                          BMI     DEC1
014044  001402                          BEQ     DEC1
014046  102401                          BVS     DEC1
014050  103401                          BCS     DEC2
014052                          DEC1:
014052  104000                          EMT                             ;DEC DID NOT SET CC'S CORRECTLY
014054  000261                  DEC2:   SEC                             ;CC=1011
014056  000244                          CLZ
014060  005300                          DEC     R0                      ;CC=0101    R0=0
014062  101002                          BHI     DEC3
014064  100401                          BMI     DEC3
014066  102001                          BVC     DEC4
014070                          DEC3:
014070  104000                          EMT                             ;DEC DID NOT SET CC'S CORRECTLY
014072  000277                  DEC4:   SCC                             ;CC=0110
014074  000251                          +CLN!CLC
014076  005300                          DEC     R0                      ;CC=1000    R0=177777
014100  101402                          BLOS    DEC5
014102  102401                          BVS     DEC5
014104  100401                          BMI     DEC6
014106                          DEC5:
014106  104000                          EMT                             ;DEC DID NOT SET CC'S CORRECTLY
014110  042700  077777          DEC6:   BIC     #77777,R0               ;R0=100000
014114  000277                          SCC                             ;CC=0101
014116  000252                          +CLN!CLV
014120  005300                          DEC     R0                      ;CC=1011    R0=77777
014122  100403                          BMI     DEC7                    ;CC=0011
014124  001402                          BEQ     DEC7
014126  102001                          BVC     DEC7
014130  103401                          BCS     TS215
014132                          DEC7:
014132  104000                          EMT                             ;DEC DID NOT SET CC'S CORRECTLY


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

```
                                ;       THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE CLR,
                                ;TST, AND SWAB INSTRUCTIONS.  THESE THREE INSTRUCTIONS ALL LEAVE
                                ;THE C AND V BITS CLEARED.  AGAIN, THE CONDITION CODES ARE PRESET,
                                ;THE INSTRUCTION EXECUTED AND THE RESULTS CHECKED WITH CONDITIONAL
                                ;BRANCH INSTRUCTIONS.  THE PROCEDURE IS REPEATED TO PRODUCE OTHER
                                ;COMBINATIONS OF CONDITION CODES.
                                ;
                                ;*************************************************************************
                                ;TEST 215        TEST CLR INSTRUCTION
                                ;*************************************************************************
014134                          TS215:
014134  000277                          SCC                             ;CC=1011
014136  000244                          CLZ
014140  005000                          CLR     R0                      ;CC=0100   R0=0
014142  100403                          BMI     CLR1
014144  102402                          BVS     CLR1
014146  103401                          BCS     CLR1
014150  001401                          BEQ     TS216
014152                          CLR1:
014152  104000                          EMT                             ;CLR DID NOT SET CC'S CORRECTLY

                                ;*************************************************************************
                                ;TEST 216        TEST TST INSTRUCTION
                                ;*************************************************************************
014154                          TS216:
014154  000277                          SCC                             ;CC=1011
014156  000244                          CLZ
014160  005700                          TST     R0                      ;CC=0100
014162  100403                          BMI     TEST1
014164  102402                          BVS     TEST1
014166  103401                          BCS     TEST1
014170  001401                          BEQ     TEST2
014172                          TEST1:
014172  104000                          EMT                             ;TEST DID NOT SET CC'S CORRECTLY
014174  005300                  TEST2:  DEC     R0                      ;MAKE R0 NEGATIVE
014176  000277                          SCC                             ;CC=0111
014200  000250                          CLN
014202  005700                          TST     R0                      ;CC=1000
014204  101402                          BLOS    TEST3
014206  102401                          BVS     TEST3
014210  100401                          BMI     TS217
014212                          TEST3:
014212  104000                          EMT                             ;TEST DID NOT SET CC'S CORRECTLY
                                ;*************************************************************************
                                ;TEST 217        TEST SWAB INSTRUCTION
                                ;*************************************************************************
014214                          TS217:
014214  012700  170000                  MOV     #170000,R0              ;R0=170000
014220  000277                          SCC                             ;CC=0111
014222  000250                          CLN
014224  000300                          SWAB    R0                      ;CC=1000   R0=360
014226  101402                          BLOS    SWB1
014230  102401                          BVS     SWB1
014232  100401                          BMI     SWB2
014234                          SWB1:
014234  104000                          EMT                             ;SWAB DID NOT SET CC'S CORRECTLY
```

```
014236  000277          SWB2:   SCC                     ;CC=1011
014240  000244                  CLZ
014242  000300                  SWAB    R0              ;CC=0100   R0=170000
014244  102403                  BVS     SWB3
014246  103402                  BCS     SWB3
014250  100401                  BMI     SWB3
014252  001401                  BEQ     TS220
014254                  SWB3:
014254  104000                  EMT                     ;

                        ;*******************************************************************************
                        ;
                        ;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE ADD AND
                        ;ADC INSTRUCTIONS.  BOTH OF THESE INSTRUCTIONS HANDLE THE C AND
                        ;V BITS IDENTICALLY.  THE PROCEDURE IS TO PRESET THE CONDITION
                        ;CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
                        ;THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
                        ;BRANCHES.  THES PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
                        ;DATA TO PRODUCE EVERY COMBINATION OF C AND V BITS.
                        ;
                        ;*******************************************************************************
                        ;TEST 220       TEST ADD INSTRUCTION
                        ;*******************************************************************************
014256                  TS220:
014256  012700  040000          MOV     #40000,R0       ;R0=40000
014262  000277                  SCC                     ;CC=1111
014264  062700  030000          ADD     #30000,R0       ;CC=0000   R0=70000
014270  101402                  BLOS    ADD1
014272  102401                  BVS     ADD1
014274  100001                  BPL     ADD2
014276                  ADD1:
014276  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY
014300  000264          ADD2:   SEZ                     ;CC=0100

014302  062700  010000          ADD     #10000,R0       ;CC=1010   40=100000
014306  101402                  BLOS    ADD3
014310  102001                  BVC     ADD3
014312  100401                  BMI     ADD4
014314                  ADD3:
014314  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY
014316  000257          ADD4:   CCC                     ;CC=1000
014320  000270                  SEN
014322  062700  100000          ADD     #100000,R0      ;CC=0111   R0=0
014326  101002                  BHI     ADD5
014330  102001                  BVC     ADD5
014332  100001                  BPL     ADD6
014334                  ADD5:
014334  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY
014336  062700  177777  ADD6:   ADD     #177777,R0      ;CC=1000   R0=177777
014342  101402                  BLOS    ADD7
014344  102401                  BVS     ADD7
014346  100401                  BMI     ADD8
014350                  ADD7:
014350  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY
014352  000277          ADD8:   SCC                     ;CC=1010
014354  000245                  +CLC!CLZ
```

```
014356  062700  000001          ADD     #1,R0           ;CC=0101    R=0
014362  102403                  BVS     ADD9
014364  103002                  BCC     ADD9
014366  100401                  BMI     ADD9
014370  001401                  BEQ     TS221
014372                  ADD9:
014372  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY


                        ;**************************************************************************
                        ;TEST 221        TEST ADC INSTRUCTION
                        ;**************************************************************************
014374                  TS221:
014374  012700  077777          MOV     #077777,R0
014400  000277                  SCC                     ;CC=0101
014402  000252                  +CLN!CLV
014404  005500                  ADC     R0              ;CC=1010
014406  101402                  BLOS    ADC1
014410  102001                  BVC     ADC1
014412  100401                  BMI     ADC2
014414                  ADC1:
014414  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY
014416  052700  077777  ADC2:   BIS     #77777,R0
014422  000277                  SCC                     ;CC=1011
014424  000244                  CLZ
014426  005500                  ADC     R0              ;CC=0101    R0=0
014430  101002                  BHI     ADC3
014432  102401                  BVS     ADC3
014434  100001                  BPL     ADC4
014436                  ADC3:
014436  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY
014440  000277          ADC4:   SCC
014442  000245                  +CLZ!CLC                        ;CC=1010
014444  005500                  ADC     R0              ;CC=0100
014446  102403                  BVS     ADC5
014450  103402                  BCS     ADC5
014452  100401                  BMI     ADC5
014454  001401                  BEQ     TS222
014456                  ADC5:
014456  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY

                        ;**************************************************************************
                        ;
                        ;       THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE NEG,
                        ;CMP, AND COM INSTRUCTIONS.  EACH OF THESE INSTRUCTIONS GENERATE
                        ;THE C AND V BITS IDENTICALLY.  THE CONDITION CODES ARE PRESET,
                        ;THE INSTRUCTIONS EXECUTED, AND THE RESULTS CHECKED WITH A SERIES
                        ;OF CONDITIONAL BRANCH INSTRUCTIONS.  THIS PROCEDURE IS REPEATED
                        ;SEVERAL TIMES WITH DIFFERENT DATA IN ORDER TO GENERATE DIFFERENT
                        ;COMBINATIONS OF THE C AND V BITS.
                        ;
                        ;**************************************************************************
                        ;TEST 222        TEST NEG INSTRUCTION
                        ;**************************************************************************
014460                  TS222:
014460  012700  000001          MOV     #1,R0
014464  000277                  SCC                     ;CC=0110
```

```
014466 000251                         +CLN!CLC
014470 005400                         NEG     R0                ;CC=1001    R0=177777
014472 103003                         BCC     NEG1
014474 102402                         BVS     NEG1
014476 001401                         BEQ     NEG1
014500 100401                         BMI     NEG2
014502                        NEG1:
014502 104000                         EMT                       ;NEG DID NOT SET CC'S CORRECTLY
014504 042700 077777          NEG2:   BIC     #77777,R0
014510 000257                         CCC                       ;CC=0100
014512 000264                         SEZ
014514 005400                         NEG     R0                ;CC=1011    R0=100000
014516 102003                         BVC     NEG3
014520 103002                         BCC     NEG3
014522 001401                         BEQ     NEG3
014524 100401                         BMI     NEG4
014526                        NEG3:
014526 104000                         EMT                       ;NEG DID NOT SET CC'S CORRECTLY
014530 005000                 NEG4:   CLR     R0
014532 000277                         SCC                       ;CC=1011
014534 000244                         CLZ
014536 005400                         NEG     R0                ;CC=0100    R0=0
014540 102403                         BVS     NEG5
014542 103402                         BCS     NEG5
014544 001001                         BNE     NEG5
014546 100001                         BPL     TS223
014550                        NEG5:
014550 104000                         EMT                       ;NEG DID NOT SET CC'S CORRECTLY


                              ;*****************************************************************************
                              ;TEST 223        TEST CMP INSTRUCTION
                              ;*****************************************************************************
014552                        TS223:
014552 012700 000005                  MOV     #5,R0
014556 000257                         CCC                       ;CC=1010
014560 000271                         +SEN!SEC
014562 022700 000005                  CMP     #5,R0             ;CC=0101
014566 101002                         BHI     CMP1
014570 102401                         BVS     CMP1
014572 100001                         BPL     CMP2
014574                        CMP1:
014574 104000                         EMT                       ;CMP DID NOT SET CC'S CORRECTLY
014576 012700 100000          CMP2:   MOV     #100000,R0
014602 000277                         SCC                       ;CC=1101
014604 000242                         CLV
014606 020027 077777                  CMP     R0,#77777         ;CC=0010
014612 101402                         BLOS    CMP3
014614 102001                         BVC     CMP3
014616 100001                         BPL     CMP4
014620                        CMP3:
014620 104000                         EMT                       ;CMP DID NOT SET CC'S CORRECTLY
014622 052700 040000          CMP4:   BIS     #40000,R0         ;R0=140000
014626 000257                         CCC                       ;CC=0100
014630 000264                         SEZ
014632 022700 040000                  CMP     #40000,R0         ;CC=1011
014636 102003                         BVC     CMP5
```

```
014640  103002                  BCC     CMP5
014642  001401                  BEQ     CMP5
014644  100401                  BMI     CMP6
014646                  CMP5:
014646  104000                  EMT                     ;CMP DID NOT SET CC'S CORRECTLY
014650  042700  040000  CMP6:   BIC     #40000,R0
014654  000277                  SCC                     ;CC=1111
014656  022700  177777          CMP     #-1,R0          ;CC=0000
014662  101402                  BLOS    CMP7
014664  102401                  BVS     CMP7
014666  100001                  BPL     TS224
014670                  CMP7:
014670  104000                  EMT                     ;CMP DID NOT SET CC'S CORRECTLY

                        ;*******************************************************************************
                        ;TEST 224       TEST COM INSTRUCTION
                        ;*******************************************************************************
014672                  TS224:
014672  012700  177777          MOV     #-1,R0
014676  000257                  CCC                     ;CC=1010
014700  000265                  +SEC!SEZ
014702  005100                  COM     R0              ;CC=0101
014704  101002                  BHI     COM1
014706  102401                  BVS     COM1
014710  100001                  BPL     TS225
014712                  COM1:
014712  104000                  EMT                     ;COM DID NOT SET CC'S CORRECTLY


                        ;*******************************************************************************
                        ;
                        ;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE SUB
                        ;AND SBC INSTRUCTIONS.  BOTH OF THESE INSTRUCTIONS HANDLE THE
                        ;C AND V BITS IDENTICALLY.  THE PROCEDURE IS TO PRESET THE CONDITION
                        ;CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
                        ;THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
                        ;BRANCHES.  THIS PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
                        ;DATA PATTERNS TO PROVIDE EVERY COMBINATION OF THE C AND V BITS.
                        ;
                        ;*******************************************************************************
                        ;TEST 225       TEST SUB INSTRUCTION
                        ;*******************************************************************************
014714                  TS225:
014714  012700  125252          MOV     #125252,R0
014720  000257                  CCC                     ;CC=1010
014722  000271                  +SEN!SEC
014724  162700  125252          SUB     #125252,R0      ;CC=0101   R0=0
014730  101002                  BHI     SUB1
014732  102401                  BVS     SUB1
014734  100001                  BPL     SUB2
014736                  SUB1:
014736  104000                  EMT                     ;SUB DID NOT SET CC'S CORRECTLY
014740  052700  100000  SUB2:   BIS     #100000,R0
014744  000277                  SCC                     ;CC=1101
014746  000242                  CLV
014750  162700  077777          SUB     #77777,R0       ;CC=0010   R0=1
```

```
014754  101402                  BLOS    SUB3
014756  102001                  BVC     SUB3
014760  100001                  BPL     SUB4
014762                  SUB3:
014762  104000                  EMT                     ;
014764  005100          SUB4:   COM     R0              ;R0=177777
014766  000277                  SCC                     ;CC=11111

014770  162700  100000          SUB     #100000,R0      ;CC=0000   R0=77777
014774  101402                  BLOS    SUB5
014776  102401                  BVS     SUB5
015000  100001                  BPL     SUB6
015002                  SUB5:
015002  104000                  EMT                     ;SUB DID NOT SET CC'S CORRECTLY
015004  000257          SUB6:   CCC                     ;CC=0100
015006  000264                  SEZ
015010  162700  140000          SUB     #140000,R0      ;CC=1011
015014  102003                  BVC     SUB7
015016  103002                  BCC     SUB7
015020  001401                  BEQ     SUB7
015022  100401                  BMI     TS226
015024                  SUB7:
015024  104000                  EMT                     ;


                        ;*****************************************************************************
                        ;TEST 226       TEST SBC INSTRUCTION
                        ;*****************************************************************************
015026                  TS226:
015026  012700  000001          MOV     #1,R0
015032  000277                  SCC                     ;CC=1011
015034  000244                  CLZ
015036  005600                  SBC     R0              ;CC=0100   R=0
015040  103403                  BCS     SBC1
015042  102402                  BVS     SBC1
015044  100401                  BMI     SBC1
015046  001401                  BEQ     SBC2
015050                  SBC1:
015050  104000                  EMT                     ;SBC DID NOT SET CC'S CORRECTLY
015052  000277          SBC2:   SCC                     ;CC=1010
015054  000245                  +CLZ!CLC
015056  005600                  SBC     R0              ;CC=0100   R=0
015060  103403                  BCS     SBC3
015062  102402                  BVS     SBC3
015064  100401                  BMI     SBC3
015066  001401                  BEQ     SBC4
015070                  SBC3:
015070  104000                  EMT                     ;SBC DID NOT SET CC'S CORRECTLY
015072  000277          SBC4:   SCC                     ;CC=0111
015074  000250                  CLN
015076  005600                  SBC     R0              ;CC=1001   R0=177777
015100  103003                  BCC     SBC5
015102  102402                  BVS     SBC5
015104  001401                  BEQ     SBC5
015106  100401                  BMI     SBC6
015110                  SBC5:
015110  104000                  EMT                     ;SBC DID NOT SET CC'S CORRECTLY
```

```
015112  042700  077777          SBC6:   BIC     #77777,R0               ;R0=100000
015116  000277                          SCC                             ;CC=1101
015120  000242                          CLV
015122  005600                          SBC     R0                      ;CC=0010
015124  101402                          BLOS    SBC7
015126  102001                          BVC     SBC7
015130  100001                          BPL     TS227
015132                          SBC7:
015132  104000                          EMT                             ;SBC DID NOT SET CC'S CORRECTLY

                                ;*******************************************************************************
                                ;
                                ;       THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF THE ROL,
                                ;ROR, ASL AND ASR INSTRUCTIONS.  SPECIAL DATA PATTERNS ARE LOADED
                                ;AND ROTATED SEVERAL TIMES FOR EACH TEST.  THE CONDITION CODES
                                ;ARE PRESET BEFORE EACH ROTATION AND THE CONDITION CODES ARE
                                ;CHECKED AFTER EACH ROTATION.  THE FINAL CHECK IN EACH TEST IS
                                ;TO VERIFY THE COMMULATIVE DATA RESULT.  THE DATA PATTERNS HAVE
                                ;BEEN SELECTED TO PRODUCE ALL COMBINATIONS OF THE C AND V BITS.
                                ;
                                ;*******************************************************************************
                                ;TEST 227       TEST ROL INSTRUCTION
                                ;*******************************************************************************
015134                          TS227:
015134  012700  144000                  MOV     #144000,R0              ;R0=144000
015140  000257                          CCC                             ;CC=0110
015142  000266                          +SEZ!SEV
015144  006100                          ROL     R0                      ;CC=1001    R0=110000
015146  103003                          BCC     ROL1
015150  102402                          BVS     ROL1
015152  001401                          BEQ     ROL1
015154  100401                          BMI     ROL2
015156                          ROL1:
015156  104000                          EMT                             ;
015160  000277                  ROL2:   SCC                             ;CC=1100
015162  000243                          +CLV!CLC
015164  006100                          ROL     R0                      ;CC=0011    R0=020000
015166  103003                          BCC     ROL3
015170  102002                          BVC     ROL3
015172  001401                          BEQ     ROL3
015174  100001                          BPL     ROL4
015176                          ROL3:
015176  104000                          EMT                             ;ROL DID NOT SET CC'S CORRECTLY
015200  000277                  ROL4:   SCC                             ;CC=0111
015202  000250                          CLN
015204  006100                          ROL     R0                      ;CC=0000    R0=040001
015206  101402                          BLOS    ROL5
015210  102401                          BVS     ROL5
015212  100001                          BPL     ROL6
015214                          ROL5:
015214  104000                          EMT                             ;ROL DID NOT SET CC'S CORRECTLY
015216  000257                  ROL6:   CCC                             ;CC=0101
015220  000265                          +SEZ!SEC
015222  006100                          ROL     R0                      ;CC=1010    R0=100003
015224  101405                          BLOS    ROL7
015226  102004                          BVC     ROL7
```

```
015230  100003                          BPL     ROL7
015232  022700  100003                  CMP     #100003,R0
015236  001401                          BEQ     TS230
015240                          ROL7:
015240  104000                          EMT                             ;ROL MALFUNCTIONED
                                ;*************************************************************************
                                ;TEST 230               TEST ROR INSTRUCTION
                                ;*************************************************************************
015242                          TS230:
015242  012700  000023                  MOV     #23,R0                  ;R0=23
015246  000277                          SCC                             ;CC=0111
015250  000250                          CLN
015252  006000                          ROR     R0                      ;CC=1001    R0=100011
015254  102403                          BVS     ROR1
015256  103002                          BCC     ROR1
015260  001401                          BEQ     ROR1
015262  100401                          BMI     ROR2
015264                          ROR1:
015264  104000                          EMT                             ;ROR DID NOT SET CC'S CORRECTLY
015266  000257                  ROR2:   CCC                             ;CC=1100
015270  000274                          +SEN!SEZ
015272  006000                          ROR     R0                      ;CC=0011    R0=040004
015274  102003                          BVC     ROR3
015276  103002                          BCC     ROR3
015300  001401                          BEQ     ROR3
015302  100001                          BPL     ROR4
015304                          ROR3:
015304  104000                          EMT                             ;ROR DID NOT SET CC'S CORRECTLY
015306  000277                  ROR4:   SCC                             ;CC=1110
015310  000241                          CLC
015312  006000                          ROR     R0                      ;CC=0000    R0=020002
015314  101403                          BLOS    ROR5
015316  102402                          BVS     ROR5
015320  001401                          BEQ     ROR5
015322  100001                          BPL     ROR6
015324                          ROR5:
015324  104000                          EMT                             ;ROR DID NOT SET CC'S CORRECTLY
015326  000257                  ROR6:   CCC                             ;CC=0101
015330  000265                          +SEC!SEZ
015332  006000                          ROR     R0                      ;CC=1010    R0=110001
015334  101402                          BLOS    ROR7
015336  102001                          BVC     ROR7
015340  100401                          BMI     TS231
015342                          ROR7:
015342  104000                          EMT                             ;ROR DID NOT PRODUCE CORRECT RESULTS
                                ;*************************************************************************
                                ;TEST 231               TEST ASL INSTRUCTION
                                ;*************************************************************************
015344                          TS231:
015344  012700  144000                  MOV     #144000,R0              ;R0=14000
015350  000257                          CCC                             ;CC=0110
015352  000271                          +SEN!SEC
015354  006300                          ASL     R0                      ;CC=1001    R0=110000
015356  103003                          BCC     ASL1
015360  102402                          BVS     ASL1
015362  001401                          BEQ     ASL1
```

```
015364  100401                   BMI     ASL2
015366                   ASL1:
015366  104000                   EMT                     ;
015370  000277           ASL2:   SCC                     ;CC=1100
015372  000243                   +CLV!CLC
015374  006300                   ASL     R0              ;CC=0011    R0=020000
015376  103003                   BCC     ASL3
015400  102002                   BVC     ASL3
015402  001401                   BEQ     ASL3
015404  100001                   BPL     ASL4
015406                   ASL3:
015406  104000                   EMT                     ;ASL DID NOT SET CC'S CORRECTLY
015410  000277           ASL4:   SCC                     ;CC=0111
015412  000250                   CLN
015414  006300                   ASL     R0              ;CC=0000    R0=040000
015416  101402                   BLOS    ASL5
015420  102401                   BVS     ASL5
015422  100001                   BPL     ASL6
015424                   ASL5:
015424  104000                   EMT                     ;ASL DID NOT SET CC'S CORRECTLY
015426  000257           ASL6:   CCC                     ;CC=0101
015430  000265                   +SEZ!SEC
015432  006300                   ASL     R0              ;CC=1010    R0=100000
015434  103406                   BCS     ASL7
015436  001405                   BEQ     ASL7
015440  102004                   BVC     ASL7
015442  100003                   BPL     ASL7
015444  022700  100000           CMP     #100000,R0
015450  001401                   BEQ     TS232
015452                   ASL7:
015452  104000                   EMT                     ;ASL MALFUNCTIONED
                         ;*****************************************************************************
                         ;TEST 232       TEST ASR INSTRUCTION
                         ;*****************************************************************************
015454                   TS232:
015454  012700  100023           MOV     #100023,R0      ;R0=100023
015460  000277                   SCC                     ;CC=0110
015462  000250                   CLN
015464  006200                   ASR     R0              ;CC=1001    RP=140011
015466  102403                   BVS     ASR1
015470  103002                   BCC     ASR1
015472  001401                   BEQ     ASR1
015474  100401                   BMI     ASR2
015476                   ASR1:
015476  104000                   EMT                     ;ASR DID NOT SET CC'S CORRECTLY
015500  042700  100000   ASR2:   BIC     #100000,R0      ;R0=40011
015504  000277                   SCC                     ;CC=1100
015506  000243                   +CLV!CLC
015510  006200                   ASR     R0              ;CC=0011    R0=020004
015512  102003                   BVC     ASR3
015514  103002                   BCC     ASR3
015516  001401                   BEQ     ASR3
015520  100001                   BPL     ASR4
015522                   ASR3:
015522  104000                   EMT                     ;ASR DID NOT SET CC'S CORRECTLY
015524  000277           ASR4:   SCC                     ;CC=1111
```

```
015526  006200                  ASR     R0              ;CC=0000   R0=010002
015530  101403                  BLOS    ASR5
015532  102402                  BVS     ASR5
015534  001401                  BEQ     ASR5
015536  100001                  BPL     ASR6
015540                  ASR5:
015540  104000                  EMT                     ;ASR DID NOT SET CC'S CORRECTLY
015542  052700  100000  ASR6:   BIS     #100000,R0      ;R0=110002
015546  000257                  CCC                     ;CC=0101
015550  000265                  +SEZ!SEC
015552  006200                  ASR     R0              ;C=1010   R0=144001
015554  101406                  BLOS    ASR7
015556  102005                  BVC     ASR7
015560  100004                  BPL     ASR7
015562  001403                  BEQ     ASR7
015564  022700  144001          CMP     #144001,R0      ;CHECK RESULT OF ASR'S
015570  001401                  BEQ     TS233
015572                  ASR7:
015572  104000                  EMT                     ;ASR DID NOT FUNCTION CORRECTLY



                        ;*************************************************************************
                        ;TEST 233       TEST RORB INSTRUCTION
                        ;*************************************************************************
015574                  TS233:
015574  112701  000004          MOVB    #4,R1   ;LOAD REGISTER
015600  000257                  CCC                     ;CLEAR ALL FLAGS
015602  106001                  RORB    R1              ;SHIFT  BYTE RIGHT
015604  106001                  RORB    R1              ;SHIFT BYTE RIGHT
015606  122701  000001          CMPB    #1,R1           ;CHECK RESULT
015612  001401                  BEQ     RORB1
015614  104000                  EMT                     ;RORB DID NOT FUNCTION CORRECTLY
015616  106001          RORB1:  RORB    R1              ;SHIFT BYTE RIGHT
015620  100403                  BMI     RORB2           ;CC=7?
015622  001002                  BNE     RORB2
015624  102001                  BVC     RORB2
015626  103401                  BCS     RORB3
015630                  RORB2:
015630  104000                  EMT                     ;RORB DID NOT SET CC'S CORRECTLY
015632  106001          RORB3:  RORB    R1              ;SHIFT BYTE RIGHT
015634  100002                  BPL     RORB4           ;CC=12
015636  101401                  BLOS    RORB4
015640  102401                  BVS     RORB5
015642                  RORB4:
015642  104000                  EMT                     ;RORB DID NOT SET CC CORRECTLY
015644  122701  000200  RORB5:  CMPB    #200,R1         ;CHECK RESULT
015650  001401                  BEQ     RORB7
015652  104000                  EMT                     ;RORB DID NOT FUNCTION CORRECTLY
015654                  RORB7:
                        ;ROTATE ODD BYTE
015654  005000                  CLR     R0              ;MAKE R0 ZERO
015656  012710  025125          MOV     #025125,(R0)    ;PUT STARTING VALUE IN LOC. 0
015662  005200                  INC     R0              ;MAKE R0 POINT TO ODD BYTE
015664  000257                  CCC                     ;CLEAR ALL CC
015666  000261                  SEC                     ;SEC CARRY BIT
```

```
015670  106010                          RORB    (R0)            ;SHIFT BYTE RIGHT
015672  100002                          BPL     RORB10          ;CC=12?
015674  101401                          BLOS    RORB10
015676  102401                          BVS     RORB11
015700                          RORB10:
015700  104000                          EMT                     ;RORB DID NOT SET CC'S CORRECTLY
015702  022737  112525  000000  RORB11: CMP     #112525,@#0     ;CHECK RESULT
015710  001401                          BEQ     RORB12
015712  104000                          EMT                     ;RORB DID NOT FUNCTION CORRECTLY
015714  106010                  RORB12: RORB    (R0)            ;SHIFT BYTE RIGHT
015716  100403                          BMI     RORB13          ;CC=3?
015720  001402                          BEQ     RORB13
015722  102001                          BVC     RORB13
015724  103401                          BCS     RORB14
015726                          RORB13:
015726  104000                          EMT                     ;RORB DID NOT SET CC CORRECTLY
015730  022737  045125  000000  RORB14: CMP     #045125,@#0     ;CHECK RESULT
015736  001401                          BEQ     TS234
015740  104000                          EMT                     ;RORB DID NOT FUNCTION CORRECTLY


                                ;*************************************************************************
                                ;TEST 234       TEST ASLB INSTRUCTION
                                ;*************************************************************************
015742                          TS234:
015742  112701  000040                  MOVB    #40,R1          ;LOAD REGISTER
015746  000257                          CCC                     ;CLEAR ALL CONDITION CODES
015750  106301                          ASLB    R1              ;SHIFT BYTE LEFT
015752  106301                          ASLB    R1              ;SHIFT BYTE LEFT
015754  100002                          BPL     ASLB2           ;CHECK CC=12
015756  101401                          BLOS    ASLB2
015760  102401                          BVS     ASLB3
015762                          ASLB2:
015762  104000                          EMT                     ;ASLB DID NOT SET CONDITION CODE CORRECTLY
015764  022701  000200          ASLB3:  CMP     #200,R1         ;CHECK RESULT
015770  001401                          BEQ     ASLB1
015772  104000                          EMT                     ;ASLB DID NOT FUNCTION CORRECTLY
015774  106301                  ASLB1:  ASLB    R1              ;SHIFT BYTE LEFT
015776  100403                          BMI     ASLB4           ;CHECK CC=7?
016000  001002                          BNE     ASLB4
016002  102001                          BVC     ASLB4
016004  103401                          BCS     TS235
016006                          ASLB4:
016006  104000                          EMT                     ;ASLB   DID NOT SET CC'S CORRECTLY


                                ;*************************************************************************
                                ;TEST 235       TEST ASRB INSTRUCTION
                                ;*************************************************************************
016010                          TS235:
016010  112701  000004                  MOVB    #4,R1           ;SET UP STARTING DATA
016014  000257                          CCC                     ;CLEAR ALL CONDITION CODES
016016  106201                          ASRB    R1              ;SKIFT BYTE RIGHT
016020  106201                          ASRB    R1              ;SHIFT BYTE RIGHT
016022  122701  000001                  CMPB    #1,R1           ;CHECK DATA
016026  001401                          BEQ     ASRB1
```

```
016030  104000                    EMT                      ;ASRB DID NOT SHIFT DATA CORRECTLY
016032  106201          ASRB1:    ASRB     R1              ;SHIFT BYTE RIGHT
016034  100403                    BMI      ASRB2           ;CHECK CONDITION CODE = 7?
016036  001002                    BNE      ASRB2
016040  102001                    BVC      ASRB2
016042  103401                    BCS      ASRB3
016044                  ASRB2:
016044  104000                    EMT                      ;ASRB DID NOT SET CC'S CORRECTLY
016046  106201          ASRB3:    ASRB     R1              ;SHIFT BYTE RIGHT
016050  103401                    BCS      ASRB4           ;CHECK CC=4
016052  001401                    BEQ      ASRB5
016054                  ASRB4:
016054  104000                    EMT                      ;ASRB DID NOT SET CC'S CORRECTLY
016056  112701  000202  ASRB5:    MOVB     #202,R1         ;PUT STARTING DATA IN REGISTER
016062  106201                    ASRB     R1              ;SHIFT BYTE RIGHT
016064  106201                    ASRB     R1              ;SHIFT BYTE RIGHT
016066  100003                    BPL      ASRB6           ;CHECK CC'S =11?
016070  001402                    BEQ      ASRB6
016072  102401                    BVS      ASRB6
016074  103401                    BCS      ASRB7
016076                  ASRB6:
016076  104000                    EMT                      ;ASRB DID NOT SET CC'S CORRECTLY
016100  122701  000340  ASRB7:    CMPB     #340,R1         ;CHECK RESULT
016104  001401                    BEQ      TS236
016106  104000                    EMT                      ;ASRB DID NOT SHIFT DATA CORRECTLY

                        ;*************************************************************************
                        ;
                        ;         THIS TEST VERIFIES THE SXT INSTRUCTION. CONDITION  CODES
                        ;ARE PRESET IN EACH OF THE TWO POSSIBLE CASES. WITH THE N-BIT SET,
                        ;THE TEST CHECKS FOR ALL ONES IN THE DESTINATION. WITH THE N-BIT
                        ;CLEAR, THE DESTINATION SHOULD CONTAIN ALL ZEROES. THE DATA
                        ;IS VERIFIED BY CONDITIONAL BRANCHES.
                        ;
                        ;*************************************************************************
                        ;TEST 236         TEST THE SXT INSTRUCTION
                        ;*************************************************************************
016110                  TS236:
016110  005000                    CLR      R0
016112  000277                    SCC                      ;SET CC=1011
016114  000244                    CLZ
016116  006700                    SXT      R0              ;TRY SXT
016120  100006                    BPL      SXT0            ;TEST CC=1001
016122  001405                    BEQ      SXT0
016124  102404                    BVS      SXT0
016126  103003                    BCC      SXT0
016130  022700  177777            CMP      #-1,R0          ;CHECK DATA RESULT
016134  001401                    BEQ      SXT1
016136                  SXT0:
016136  104000                    EMT                      ;RESULTS OF SXT INCORRECT
016140  005000          SXT1:     CLR      R0              ;R0=0
016142  005010                    CLR      (R0)            ;LOC. 0=0
016144  005110                    COM      (R0)            ;LOC. 0=177777
016146  000257                    CCC                      ;SET CC=0110
016150  000266                    +SEZ!SEV
016152  006710                    SXT      (R0)
```

```
016154  001005              BNE     SXT2            ;TEST CC=0100
016156  103404              BCS     SXT2
016160  102403              BVS     SXT2
016162  100402              BMI     SXT2
016164  005710              TST     (R0)
016166  001401              BEQ     TS237
016170                SXT2:
016170  104000              EMT                     ;RESULTS OF SXT INCORRECT
                      ;*************************************************************************************
                      ;
```

```
                                ;       THIS TEST VERIFIES THE XOR INSTRUCTION. UNIQUE PATTERNS
                                ;OF ONES AND ZEROES ARE MOVED TO DATA REGISTERS R0 AND R1.
                                ;AFTER THE FIRST XOR INSTRUCTION R0=36146. AN XOR IS THEN
                                ;EXECUTED WITH THIS NEW VALUE AND THE CONTENTS OF R1 TO
                                ;REPRODUCE THE ORIGINAL VALUE IF R0=31525.
                                ;
                                ;*******************************************************************************
                                ;TEST 237        TEST THE XOR INSTRUCTION
                                ;*******************************************************************************
016172                          TS237:
016172  012700  007463                  MOV     #7463,R0        ;SET UP R0
016176  012701  031525                  MOV     #31525,R1       ;SET UP R1
016202  000277                          SCC                     ;SET CC=1110
016204  000241                          CLC
016206  074100                          XOR     R1,R0           ;TRY XOR
016210  101406                          BLOS    XOR1            ;CC=0000?
016212  102405                          BVS     XOR1
016214  001404                          BEQ     XOR1
016216  100403                          BMI     XOR1
016220  022700  036146                  CMP     #36146,R0       ;DATA RESULT CORRECT?
016224  001401                          BEQ     XOR2
016226                          XOR1:
016226  104000                          EMT                     ;
016230  010104                  XOR2:   MOV     R1,R4
016232  000261                          SEC                     ;CC=1110
016234  000241                          CLC
016236  074400                          XOR     R4,R0           ;TRY XOR MODE 0,0
016240  101406                          BLOS    XOR3            ;CC=0000?
016242  102405                          BVS     XOR3
016244  001404                          BEQ     XOR3
016246  100403                          BMI     XOR3
016250  022700  007463                  CMP     #7463,R0
016254  001401                          BEQ     TS240
016256                          XOR3:
016256  104000                          EMT                     ;RESULT OF XOR INCORRECT
                                ;*******************************************************************************
                                ;
                                ;
                                ;       THIS TEST VERIFIES THE SOB INSTRUCTION. R4 IS USED AS A
                                ;COUNTER WHILE R0 IS THE ADDRESS REGISTER. CONDITIONAL
                                ;BRANCHES ARE USED TO VERIFY PROPER TRANSFER OF CONTROL
                                ;WHILE R4 IS CHECKED TO INSURE PROPER DECREMENTING OF R0.
                                ;
                                ;*******************************************************************************
                                ;TEST 240        TEST SOB INSTRUCTION
                                ;*******************************************************************************
016260                          TS240:
016260  012700  000525                  MOV     #525,R0
016264  010004                          MOV     R0,R4
016266  000277                          SCC                     ;SET CC=1111
016270  101002                  SOB1:   BHI     SOB2            ;CC=1111?
016272  100001                          BPL     SOB2
016274  102401                          BVS     SOB3
016276                          SOB2:
016276  104000                          EMT                     ;
016300  005304                  SOB3:   DEC     R4              ;COUNT ITERATIONS
016302  000277                          SCC                     ;CC=1111
```

```
016304  077007                  SOB     R0,SOB1         ;DO SOB W/ R0
016306  101004                  BHI     SOB4            ;CHECK CC=1111
016310  100003                  BPL     SOB4
016312  102002                  BVC     SOB4
016314  005704                  TST     R4              ;ITERATION COUNT OK?
016316  001401                  BEQ     TS241
016320                  SOB4:
016320  104000                  EMT                     ;INCORRECT # OF BRANCHES OR CC'S CHANGED
                        ;*******************************************************************************
                        ;
                        ;
                        ;       THIS TEST VERIFIES THE MARK INSTRUCTION. THE EFFECTS
                        ;OF THE MARK INSTRUCTION ARE SIMULATED BY THE PROGRAM INSTRUCTIONS.
                        ;THE CONTENTS OF R5 AND THE STACK POINTER ARE CHECKED AFTER EACH
                        ;OF THE TWO ROUTINES IN THE TEST.
                        ;
                        ;*******************************************************************************
                        ;TEST 241       TEST MARK INSTRUCTION
                        ;*******************************************************************************
016322                  TS241:
016322  012706  001000          MOV     #STBOT,SP
016326  012746  125252          MOV     #125252,-(SP)   ;PUT R5 VALUE ON STACK
016332  162706  000074          SUB     #74,SP          ;EFFECTIVELY PUT 36 ARGUMENTS ON STACK
016336  012705  016354          MOV     #MRK1,R5        ;SET NEW PC IN R5
016342  012746  006436          MOV     #6436,-(SP)     ;PUT MARK 36 INST. ON STACK
016346  000277                  SCC                     ;SET CC=1111
016350  000116                  JMP     (SP)            ;XFER CONTL TO MARK 36 INST. ON STACK
016352  104000                  EMT                     ;MARK INST. SHOULD HAVE JUMPED TO MRK1
016354  101010          MRK1:   BHI     MRK2            ;TEST CC UNAFFECTED
016356  100007                  BPL     MRK2            ;IE. CC=1111
016360  102006                  BVC     MRK2
016362  020527  125252          CMP     R5,#125252      ;CHECK R5 RESTORED FROM STACK
016366  001003                  BNE     MRK2
016370  022706  001000          CMP     #STBOT,R6       ;CHECK STACK POINTER READJUSTED CORRECTLY.
016374  001401                  BEQ     MRK3
016376                  MRK2:
016376  104000                  EMT                     ;RESULTS OF MARK INCORRECT
016400  012746  052525  MRK3:   MOV     #52525,-(SP)
016404  012746  006400          MOV     #6400,-(SP)     ;PUT MARK 0 INST. ON STACK
016410  010605                  MOV     SP,R5           ;SET ADDR. OF MARK INST. IN R5
016412  004737  016422          JSR     PC,@#MRK4       ;DO JSR
016416  000137  016426          JMP     @#MRK5          ;
016422  000205          MRK4:   RTS     R5              ;DO RTS WITH R5 TO MARK INST ON STACK
016424  104000                  EMT                     ;RTS,MARK SEQUENCE FAILED
016426  022706  001000  MRK5:   CMP     #STBOT,R6       ;STACK ADJUSTED CORRECTLY
016432  001003                  BNE     MRK6            ;IF NOT:  BR
016434  022705  052525          CMP     #52525,R5       ;CHECK IF R5 RESTORED FROM STACK
016440  001401                  BEQ     TS242
016442                  MRK6:
016442  104000                  EMT                     ;RESULTS OF MARK INCORRECT
        177776                  PS=177776
                        ;*******************************************************************************
                        ;
                        ;
                        ;       THESE NEXT SEVEN TESTS VERIFY THE MTPS INSTRUCTION IN ALL
                        ;       MODES. THE PSW IS DEFINED BY AN EQUATE STATEMENT BEFORE THE
                        ;       FIRST MTPS TEST. IN EACH TEST A PATTERN OF ONES AND
                        ;       ZEROES IS SET IN A DATA REGISTER AND MOVED TO THE PSW.
```

```
                                        ;       THE DATA IN THE PSW, AND THE DATA REGISTER ADDRESS,
                                        ;       ARE CHECKED TO VERIFY PROPER EXECUTION OF THE INSTRUCTION.
                                        ;

                                        ;***********************************************************************
                                        ;TEST 242        TEST MTPS INSTRUCTION
                                        ;***********************************************************************
016444                                  TS242:
016444  012700  000377                          MOV     #377,R0
016450  000257                                  CCC
016452  106400                                  MTPS    R0
016454  022767  000357  161314                  CMP     #357,PS
016462  001401                                  BEQ     MTPS1
016464  104000                                  EMT                     ;MTPS FAILED
016466  005000                          MTPS1:  CLR     R0
016470  005010                                  CLR     (R0)
016472  000277                                  SCC                     ;CC=1111
016474  106410                                  MTPS    (R0)            ;TRY MTPS MODE 1
016476  100403                                  BMI     MTPS1A          ;CHECK PS
016500  102402                                  BVS     MTPS1A
016502  103401                                  BCS     MTPS1A
016504  001001                                  BNE     TS243
016506                                  MTPS1A:
016506  104000                                  EMT                     ;MTPS FAILED

                                        ;***********************************************************************
                                        ;TEST 243        TEST MTPS MODE 2
                                        ;***********************************************************************
016510                                  TS243:
016510  005000                                  CLR     R0              ;R0=0
016512  012710  177777                          MOV     #-1,(R0)        ;LOC. 0=-1
016516  005037  177776                          CLR     @#PS            ;PS=0
016522  106420                                  MTPS    (R0)+           ;TRY MTPS W/MODE 2
016524  022737  000357  177776                  CMP     #357,@#PS       ;CHECK DATA
016532  001401                                  BEQ     MTPS2
016534  104000                                  EMT                     ;DEST. DATA INCORRECT
016536  022700  000001                  MTPS2:  CMP     #1,R0           ;CHECK DEST. REGISTER.
016542  001401                                  BEQ     TS244
016544  104000                                  EMT                     ;DEST REGISTER NOT INCREMENTED BY 1

                                        ;***********************************************************************
                                        ;TEST 244        TEST MTPS MODE 3
                                        ;***********************************************************************
016546                                  TS244:
016546  012700  000402                          MOV     #402,R0         ;R0=402
016552  005010                                  CLR     (R0)            ;LOC. 402=0
016554  012737  052652  000000                  MOV     #52652,@#0      ;LOC. 0=52652
016562  005037  177776                          CLR     @#PS            ;PS=0
016566  106430                                  MTPS    @(R0)+          ;TRY MTPS W/MODE 3
016570  022737  000252  177776                  CMP     #252,@#PS       ;CHECK DEST. DATA
016576  001401                                  BEQ     MTPS3
016600  104000                                  EMT                     ;DEST. DATA INCORRECT
016602  022700  000404                  MTPS3:  CMP     #404,R0         ;CHECK MODE 3 REGISTER.
016606  001401                                  BEQ     TS245
016610  104000                                  EMT                     ;MODE 3 REGISTER INCORRECT

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

```
                                        ;TEST 245       TEST MTPS MODE 4
                                        ;*******************************************************************************
016612                                  TS245:
016612  012700  000001                          MOV     #1,R0           ;R0=1
016616  012737  125125  000000                  MOV     #125125,@#0     ;LOC. 0 = 125125
016624  005037  177776                          CLR     @#PS            ;PS=0
016630  106440                                  MTPS    -(R0)           ;TRY MTPS W/MODE 4
016632  022737  000105  177776                  CMP     #105,@#PS       ;CHECK DEST. DATA
016640  001401                                  BEQ     MTPS4
016642  104000                                  EMT                     ;DEST. DATA INCORRECT
016644  005700                          MTPS4:  TST     R0              ;CHECK MODE 4 REGISTER
016646  001401                                  BEQ     TS246
016650  104000                                  EMT                     ;MODE 4 REGISTER NOT DECREMENTED BY 1

                                        ;*******************************************************************************
                                        ;TEST 246       TEST MTPS MODE 5
                                        ;*******************************************************************************
016652                                  TS246:
016652  012700  000404                          MOV     #404,R0         ;R0=404
016656  012737  177400  000000                  MOV     #177400,@#0     ;LOC. 0=177400
016664  000277                                  SCC                     ;SET ALL COND. CODES
016666  106450                                  MTPS    @-(R0)          ;TRY MTPS W/MODE 5
016670  005737  177776                          TST     @#PS            ;CHECK DEST. DATA.
016674  001401                                  BEQ     MTPS5
016676  104000                                  EMT                     ;DESTINATION DATA INCORRECT
016700  022700  000402                  MTPS5:  CMP     #402,R0         ;CHECK MODE 5 REGISTER
016704  001401                                  BEQ     TS247
016706  104000                                  EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2

                                        ;*******************************************************************************
                                        ;TEST 247       TEST MTPS MODE 6
                                        ;*******************************************************************************
016710                                  TS247:
016710  012737  052652  000000                  MOV     #52652,@#0      ;LOC. 0=52652
016716  012700  000406                          MOV     #406,R0         ;R0=406
016722  005037  177776                          CLR     @#PS            ;PS=0
016726  106460  177372                          MTPS    -406(R0)        ;TRY MTPS W/MODE 6
016732  022737  000252  177776                  CMP     #252,@#PS       ;CHECK DEST. DATA
016740  001401                                  BEQ     MTPS6
016742  104000                                  EMT                     ;DEST. DATA INCORRECT
016744  022700  000406                  MTPS6:  CMP     #406,R0         ;CHECK MODE 6 REGISTER
016750  001401                                  BEQ     TS250
016752  104000                                  EMT                     ;MODE 6 REGISTER MODIFIED

                                        ;*******************************************************************************
                                        ;TEST 250       TEST MTPS MODE 7
                                        ;*******************************************************************************
016754                                  TS250:
016754  012737  052652  000000                  MOV     #52652,@#0      ;LOC. 0=52652
016762  012700  000410                          MOV     #410,R0         ;R0=410
016766  005037  177776                          CLR     @#PS            ;PS=0
016772  106470  177776                          MTPS    @-2(R0)         ;TRY MTPS W/MODE 7
016776  022737  000105  177776                  CMP     #105,@#PS       ;CHECK DEST. DATA
017004  001401                                  BEQ     MTPS7
017006  104000                                  EMT                     ;DESTINATION DATA INCORRECT
017010  022700  000410                  MTPS7:  CMP     #410,R0         ;CHECK MODE 7 REGISTER
```

```
017014  001401                          BEQ     TS251
017016  104000                          EMT                     ;MODE 7 REGISTER MODIFIED

                                ;*******************************************************************************
                                ;
                                ;       THESE NEXT SEVEN TESTS VERIFY THE MFPS INSTRUCTION IN ALL
                                ;MODES. IN EACH TEST, A PATTERN OF ONES AND ZEROES IS MOVED TO THE
                                ;PSW, AND AN MFPS INSTRUCTION MOVES THE DATA TO A LOCATION SETUP
                                ;BY R0, EITHER DIRECTLY OR INDIRECTLY. CONDITIONAL BRANCHES ARE
                                ;USED TO CHECK PROPER ADDRESSING AND DATA.
                                ;
                                ;*******************************************************************************
                                ;TEST 251       TEST MFPS INSTRUCTION
                                ;*******************************************************************************
017020                          TS251:
017020  012737  000377  177776          MOV     #377,@#PS
017026  106700                          MFPS    R0
017030  022700  177757                  CMP     #177757,R0
017034  001401                          BEQ     MFPS1
017036  104000                          EMT                     ;MFPS FAILED

017040  005000                  MFPS1:  CLR     R0
017042  012737  177777  000000          MOV     #-1,@#0
017050  005037  177776                  CLR     @#PS
017054  106710                          MFPS    (R0)
017056  105737  000000                  TSTB    @#0
017062  001401                          BEQ     TS252
017064  104000                          EMT                     ;MFPS FAILED

                                ;*******************************************************************************
                                ;TEST 252       TEST MFPS MODE 2
                                ;*******************************************************************************
017066                          TS252:
017066  005000                          CLR     R0              ;R0=0
017070  005010                          CLR     (R0)            ;LOC. 0=0
017072  012737  000377  177776          MOV     #377,@#PS       ;SET PS=357
017100  106720                          MFPS    (R0)+           ;TRY MFPS W/MODE 2
017102  103003                          BCC     MFPS2A          ;BR TO ERROR IF C BIT CLEAR
017104  102402                          BVS     MFPS2A          ;BR TO ERROR IF V BIT SET
017106  001401                          BEQ     MFPS2A          ;BR TO ERROR IF Z BIT SET
017110  100401                          BMI     MFPS2B
017112                          MFPS2A:
017112  104000                          EMT                     ;COND. CODES INCORRECT
017114  022737  000357  000000  MFPS2B: CMP     #357,@#0        ;CHECK DEST. DATA
017122  001401                          BEQ     MFPS2C
017124  104000                          EMT                     ;DEST. DATA INCORRECT
017126  022700  000001          MFPS2C: CMP     #1,R0           ;CHECK MODE Z REGISTER
017132  001401                          BEQ     TS253
017134  104000                          EMT                     ;MODE 2 REGISTER NOT INCREMENTED 1

                                ;*******************************************************************************
                                ;TEST 253       TEST MFPS MODE 3
                                ;*******************************************************************************
017136                          TS253:
017136  012700  000406                  MOV     #406,R0         ;R0=406
017142  005037  000000                  CLR     @#0             ;LOC. 0=0
```

```
017146  012737  000252  177776          MOV     #252,@#PS       ;PS=252
017154  106730                          MFPS    @(R0)+          ;TRY MFPS WITH MODE 3
017156  103403                          BCS     MFPS3A          ;BR TO ERROR IF C-BIT SET
017160  102402                          BVS     MFPS3A          ;BR TO ERROR IF V-BIT SET
017162  001401                          BEQ     MFPS3A          ;BR TO ERROR IF Z-BIT SET
017164  100401                          BMI     MFPS3B
017166                          MFPS3A:
017166  104000                          EMT                     ;CONDITION CODES INCORRECT
017170  022737  125000  000000  MFPS3B: CMP     #125000,@#0     ;CHECK DEST. DATA
017176  001401                          BEQ     MFPS3C
017200  104000                          EMT                     ;DEST DATA INCORRECT
017202  020027  000410          MFPS3C: CMP     R0,#410         ;CHECK MODE 3 REGISTER.
017206  001401                          BEQ     TS254
017210  104000                          EMT                     ;MODE 3 REGISTER NOT INCREMENTED BY 2

                                ;*****************************************************************
                                ;TEST 254       TEST MFPS MODE 4
                                ;*****************************************************************
017212                          TS254:
017212  012700  000002                  MOV     #2,R0           ;R0=2
017216  005037  000000                  CLR     @#0             ;LOC. 0=0
017222  012737  000125  177776          MOV     #125,@#PS       ;PS=125
017230  106740                          MFPS    -(R0)           ;TRY MFPS W/MODE 4
017232  103003                          BCC     MFPS4A          ;BR TO ERROR IF C-BIT CLEAR
017234  102402                          BVS     MFPS4A          ;BR TO ERROR IF V-BIT SET
017236  001401                          BEQ     MFPS4A          ;BR TO ERROR IF Z-BIT SET
017240  100001                          BPL     MFPS4B
017242                          MFPS4A:
017242  104000                          EMT                     ;COND. CODES INCORRECT
017244  022737  042400  000000  MFPS4B: CMP     #42400,@#0      ;CHECK DEST. DATA
017252  001401                          BEQ     MFPS4C
017254  104000                          EMT                     ;DEST. DATA INCORRECT
017256  020027  000001          MFPS4C: CMP     R0,#1           ;CHECK MODE 4 REGISTER
017262  001401                          BEQ     TS255
017264  104000                          EMT                     ;MODE 4 REGISTER NOT DECREMENTED BY 1

                                ;*****************************************************************
                                ;TEST 255       TEST MFPS MODE 5
                                ;*****************************************************************
017266                          TS255:
017266  012700  000410                  MOV     #410,R0         ;R0=410
017272  012737  177777  000000          MOV     #-1,@#0         ;LOC. 0=-1
017300  005037  177776                  CLR     @#PS            ;PS=0
017304  106750                          MFPS    @-(R0)          ;TRY MFPS W/MODE 5
017306  103403                          BCS     MFPS5A          ;BR TO ERROR IF C-BIT SET
017310  102402                          BVS     MFPS5A          ;BR TO ERROR IF V-BIT SET
017312  100401                          BMI     MFPS5A          ;BR TO ERROR IF N-BIT SET
017314  001401                          BEQ     MFPS5B
017316                          MFPS5A:
017316  104000                          EMT                     ;COND. CODES INCORRECT
017320  022737  000377  000000  MFPS5B: CMP     #377,@#0        ;CHECK DEST. DATA
017326  001401                          BEQ     MFPS5C
017330  104000                          EMT                     ;DEST DATA INCORRECT
017332  020027  000406          MFPS5C: CMP     R0,#406         ;CHECK MODE 5 REGISTER
017336  001401                          BEQ     TS256
017340  104000                          EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2
```

```

                                        ;*************************************************************************
                                        ;TEST 256        TEST MFPS MODE 6
                                        ;*************************************************************************
017342                                  TS256:
017342  012700  000401                          MOV     #401,R0                 ;R0=410
017346  005037  000000                          CLR     @#0                     ;LOC. 0=0
017352  012737  000252  177776                  MOV     #252,@#PS               ;PS=252
017360  106760  177377                          MFPS    -401(R0)                ;TRY MFPS W/MODE 6
017364  102403                                  BVS     MFPS6A                  ;BR TO ERROR IF V-BIT SET
017366  103402                                  BCS     MFPS6A                  ;BR TO ERROR IF C-BIT SET
017370  001401                                  BEQ     MFPS6A                  ;BR TO ERROR IF Z-BIT SET
017372  100401                                  BMI     MFPS6B
017374                                  MFPS6A:
017374  104000                                  EMT                             ;COND. CODES INCORRECT
017376  022737  000252  000000          MFPS6B: CMP     #252,@#0                ;CHECK DEST. DATA
017404  001401                                  BEQ     MFPS6C
017406  104000                                  EMT                             ;DEST. DATA INCORRECT
017410  022700  000401                  MFPS6C: CMP     #401,R0                 ;CHECK DEST. REGISTER
017414  001401                                  BEQ     TS257
017416  104000                                  EMT                             ;DEST. DATA INCORRECT

                                        ;*************************************************************************
                                        ;TEST 257        TEST MFPS MODE 7
                                        ;*************************************************************************
017420                                  TS257:
017420  012700  000777                          MOV     #777,R0                 ;R0=777
017424  005037  000000                          CLR     @#0                     ;LOC. 0=0
017430  012737  000125  177776                  MOV     #125,@#PS               ;PS=125
017436  106770  177407                          MFPS    @-371(R0)               ;TRY MFPS W/MODE 7
017442  102403                                  BVS     MFPS7A                  ;BR TO ERROR IF V-BIT SET
017444  103002                                  BCC     MFPS7A                  ;BR TO ERROR IF C-BIT SET
017446  001401                                  BEQ     MFPS7A                  ;BR TO ERROR IF Z-BIT SET
017450  100001                                  BPL     MFPS7B
017452                                  MFPS7A:
017452  104000                                  EMT                             ;CONDITION CODE INCORRECT
017454  022737  042400  000000          MFPS7B: CMP     #42400,@#0              ;CHECK DESTINATION DATA
017462  001401                                  BEQ     MFPS7C
017464  104000                                  EMT                             ;DEST. DATA INCORRECT
017466  022700  000777                  MFPS7C: CMP     #777,R0                 ;CHECK MODE 7 REGISTER
017472  001401                                  BEQ     TS260
017474  104000                                  EMT                             ;MODE 7 REGISTER MODIFIED

                                        ;*************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES THAT RESET DOES NOT CLEAR THE PSW.
                                        ;THE PSW IS LOADED WITH ONES, A RESET IS ISSUED, AND THE
                                        ;CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT THEY HAVE NOT
                                        ;CHANGED. THIS TEST IS EXECUTED ONLY ONCE EVERY 240 (DECIMAL)
                                        ;ITERATIONS OF PROGRAM.
                                        ;
                                        ;*************************************************************************
                                        ;TEST 260        TEST THAT RESET DOES NOT CLEAR PSW
                                        ;*************************************************************************
017476                                  TS260:
017476  032737  000001  001020                  BIT     #1, @#$ENV              ;ARE WE RUNNING UNDER APT
```

```
017504  001403                          BEQ     70$             ;IF NO THEN DO TEST
017506  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
017512  001011                          BNE     TS261           ;IF NO THEN SHIP TO NEXT TEST
017514                          70$:
017514  012737  000357  177776          MOV     #357,@#PS               ;MOV ONES TO PSW
017522  000005                          RESET                           ;
017524  022737  000357  177776          CMP     #357,@#PS               ;PSW CORRECT?
017532  001401                          BEQ     TS261
017534  104000                          EMT                     ;RESET ALTERED PSW
017536                          REST:

                                ;*******************************************************************************
                                ;
                                ;       THE FOLLOWING TEST CHECKS THE INDEPENDENT FUNCTIONING OF BASIC
                                ;DATA PATH COMPONENTS WITH USER MODE SET.
                                ;
                                ;*******************************************************************************
                                ;TEST 261       TEST USER MODE R6 CAN HOLD A ONE IN EVERY POSITION
                                ;*******************************************************************************
017536                          TS261:
017536  052767  140000  160232          BIS     #USRM,PS                ;SET USER MODE
017544  012706  000001                  MOV     #1,R6                   ;SET BIT0
017550  000241                          CLC                             ;CLEAR C-BIT
017552  006106                  USP1:   ROL     R6                      ;ROTATE 1 POSITION
017554  103376                          BCC     USP1                    ;BR IF NOT ALL DONE
017556  001404                          BEQ     USP1A                   ;BR IF NO BITS PICKED
017560  042767  140000  160210          BIC     #USRM,PS                ;CLEAR USER MODE
017566  104000                          EMT                     ;USER MODE R6 PICKED A BIT
017570  042767  140000  160200  USP1A:  BIC     #USRM,PS                ;CLEAR USER MODE


                                ;*******************************************************************************
                                ;
                                ;       THIS TEST CHECKS THE INDEPENDENT FUNCTIONING OF THE USER
                                ;AND KERNEL MODE R6'S. R6 IS SETUP AND ADDRESSED IN EACH
                                ;OF THE TWO MODES TO VERIFY THAT THE TWO R6'S ARE INDEPENDENT
                                ;OF EACH OTHER.
                                ;
                                ;*******************************************************************************
                                ;TEST 262       TEST INDEPENDENCE OF USER AND KERNEL MODE R6'S
                                ;*******************************************************************************
017576                          TS262:
017576  052767  140000  160172          BIS     #USRM,PS                ;SET USER MODE
017604  012706  177777                  MOV     #-1,R6                  ;SET USER R6 TO ALL ONES
017610  022706  177777                  CMP     #-1,R6                  ;READ AND CHECK USER R6
017614  001404                          BEQ     USP2                    ;BR IF NO ERROR
017616  042767  140000  160152          BIC     #USRM,PS                ;CLEAR USER MODE
017624  104000                          EMT                     ;USER R6 WILL NOT HOLD ALL ONES
017626  042767  140000  160142  USP2:   BIC     #USRM,PS                ;SET KERNEL MODE
017634  022706  177777                  CMP     #-1,R6                  ;KERNEL MODE R6 ADDR. FROM USER MODE?>>
017640  001001                          BNE     USP3
017642  104000                          EMT                     ;DUAL ADDRESSING ERROR USER/KERNEL R6
017644  005006                  USP3:   CLR     R6                      ;CLEAR KERNEL MODE SP
017646  052767  140000  160122          BIS     #USRM,PS                ;SET USER MODE
017654  022706  177777                  CMP     #-1,R6                  ;CHECK USER R6 NOT ADDR. FROM KERNEL MODE
017660  042767  140000  160110          BIC     #USRM,PS                ;CLEAR USER MODE
```

```
017666  001401                          BEQ     USP4                    ;BR IF NO ERROR
017670  104000                          EMT                     ;DUAL ADDRESSING ERROR OR SEQUENCE ERROR
017672  012706  001000          USP4:   MOV     #STBOT,R6               ;RESTORE SP USER
017676  042767  140000  160072          BIC     #USRM,PS                ;SET KERNEL MODE
017704  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP KERNEL

                                ;*************************************************************************
                                ;
                                ;       THESE NEXT TWO TESTS VERIFY MFPI AND MTPI INSTRUCTIONS
                                ;WITH R6 IN MODE 0.
                                ;
                                ;*************************************************************************
                                ;TEST 263       TEST MFPI WITH R6 IN MODE 0
                                ;*************************************************************************
017710                          TS263:
017710  012706  001000                  MOV     #STBOT,R6               ;INITIALIZE KERNEL STACK POINTER
017714  012767  140000  160054          MOV     #USRM,PS                ;SET USER MODE.PREVIOUS KERNEL
017722  012706  000600                  MOV     #USESTK,R6              ;INITIALIZE USER STACK POINTER
017726  006506                          MFPI    R6                      ;TRY MFPI WITH MODE 0
017730  022767  140000  160040          CMP     #140000,PS              ;CHECK PSW
017736  001404                          BEQ     MFPI0                   ;BR IF NO ERROR
017740  042767  140000  160030          BIC     #USRM,PS                ;CLEAR USER MODE
017746  104000                          EMT                     ;INCORRECT PSW FROM MFPI
017750  042767  140000  160020  MFPI0:  BIC     #USRM,PS                ;CLEAR USER MODE
017756  022767  001000  160612          CMP     #STBOT,USESTK-2         ;CHECK DATA ON STACK
017764  001401                          BEQ     MFPI0A                  ;BR IF NO ERROR
017766  104000                          EMT                     ;INCORRECT DATA FROM MFPI
017770                          MFPI0A:

                                ;*************************************************************************
                                ;TEST 264       TEST MTPI WITH R6 IN MODE 0
                                ;*************************************************************************
017770                          TS264:
017770  005067  160002                  CLR     PS                      ;SET KERNEL MODE
017774  005006                          CLR     R6                      ;INITIALIZE KERNEL R6
017776  012767  140000  157772          MOV     #USRM,PS                ;SET USER MODE/PREVIOUS KERNEL
020004  012706  000600                  MOV     #USESTK,R6              ;INITIALIZE USER STACK POINTER
020010  012746  001000                  MOV     #STBOT,-(R6)            ;SET UP TARGET DATA
020014  006606                          MTPI    R6                      ;TRY MODE 0 MTPI
020016  022767  140000  157752          CMP     #USRM,PS                ;CHECK PSW
020024  001404                          BEQ     MTPI0                   ;BR IF NO ERROR
020026  042767  140000  157742          BIC     #USRM,PS                ;CLEAR USER MODE
020034  104000                          EMT                     ;PS INCORRECT FOLLOWING MTPI
020036  005067  157734          MTPI0:  CLR     PS                      ;SET KERNEL MODE
020042  020627  001000                  CMP     R6,#STBOT               ;CHECK TARGET DATA
020046  001401                          BEQ     TS265
020050  104000                          EMT                     ;DATA INCORRECT FOLLOWING MTPI


                                ;*************************************************************************
                                ;
                                ;THE FOLLOWING TEST VERIFIES THAT NO DUAL ADDRESSING OF THE GENERAL
                                ;REGISTERS OCCURS. ALL REGISTERS ARE CLEARED, AND A UNIQUE BIT IS SET
                                ;IN EACH. CMP INSTRUCTIONS CHECK THAT ONLY ONE BIT IS SET IN EACH
                                ;REGISTER.
```

```
                                        ;
                                        ;***********************************************************************
                                        ;TEST 265       DUAL REGISTER ADDRESSING TEST
                                        ;***********************************************************************
020052                                  TS265:
020052  005000                          BITCLR: CLR     R0              ;INITIALIZE ALL REGISTERS
020054  005001                                  CLR     R1
020056  005002                                  CLR     R2
020060  005003                                  CLR     R3
020062  005004                                  CLR     R4
020064  005005                                  CLR     R5
020066  005006                                  CLR     R6
020070  052700  000001                  BITSET: BIS     #1,R0           ;SET R0=1
020074  052701  000002                          BIS     #2,R1           ;R1=2
020100  052702  000004                          BIS     #4,R2           ;R2=4
020104  052703  000010                          BIS     #10,R3          ;R3=10
020110  052704  000020                          BIS     #20,R4          ;R4=20
020114  052705  000040                          BIS     #40,R5          ;R5=40
020120  052706  000100                          BIS     #100,R6         ;R6=100
020124  022706  000100                  BITCHK: CMP     #100,R6         ;TEST THAT NO DUAL ADDRESSING OCCURRED
020130  001022                                  BNE     DAERR           ;BR TO ERROR HALT IF ANY OTHER BITS ARE SET
020132  022705  000040                          CMP     #40,R5
020136  001017                                  BNE     DAERR
020140  022704  000020                          CMP     #20,R4
020144  001014                                  BNE     DAERR
020146  022703  000010                          CMP     #10,R3
020152  001011                                  BNE     DAERR
020154  022702  000004                          CMP     #4,R2
020160  001006                                  BNE     DAERR
020162  022701  000002                          CMP     #2,R1
020166  001003                                  BNE     DAERR
020170  022700  000001                          CMP     #1,R0
020174  001401                                  BEQ     BITCON
020176                                  DAERR:
020176  104000                                  EMT                     ;DUAL ADDRESSING ERROR
020200  012702  001004                  BITCON: MOV     #$TESTN,R2      ;RESTORE POINTER

                                        ;***********************************************************************
                                        ;       THIS TEST VERIFIES THAT THE UPPER BYTE OF THE PSW IS NOT AFFECTED
                                        ;WHEN THE PRIORITY LEVEL OR CC'S ARE CHANGED. ALL BITS ARE
                                        ;INITIALLY SET IN THE PSW, AND THE LOW BYTE IS CLEARED. A BIT
                                        ;INSTRUCTION VERIFIES THE DATA.
                                        ;
                                        ;***********************************************************************
                                        ;TEST 266       TEST BYTE INSTRUCTION ON PSW
                                        ;***********************************************************************
020204                                  TS266:
020204  052737  170357  177776                  BIS     #170357,@#PS    ;SET ALL POSSIBLE BITS IN PSW
020212  105037  177776                          CLRB    @#PS            ;CLR PR LEVEL AND CC'S
020216  013700  177776                          MOV     @#PS,R0         ;COPY CONTENTS OF PSW
020222  032700  170000                          BIT     #170000,R0      ;TEST THAT UPPER BYTE IS UNAFFECTED
020226  001003                                  BNE     BTCON           ;CONTINUE IF OK
020230  005037  177776                  BTERR:  CLR     @#PS            ;RETURN TO KERNEL MODE
020234  104000                                  EMT                     ;BYTE INSTRUCTION ALTERED PSW
020236  005037  177776                  BTCON:  CLR     @#PS            ;RETURN TO KERNEL MODE
```

```
                                         ;*****************************************************************************
                                         ;
                                         ;       THIS TEST VERIFIES THAT A JMP INSTRUCTION DOES NOT ALTER THE
                                         ;CONDITION CODES IN THE PSW. THE CC'S ARE PRESET,THE JMP IS
                                         ;EXECUTED, AND CONDITIONAL BRANCHES VERIFY THE STATE OF THE CC'S.
                                         ;
                                         ;*****************************************************************************
                                         ;TEST 267        TEST THAT JMP INSTRUCTION DOES NOT AFFECT CONDITION CODES
                                         ;*****************************************************************************
020242                                   TS267:
020242  000277                                   SCC
020244  000252                                   +CLN!CLV                        ;CC=0101
020246  000167  000000                           JMP     JMPT                    ;JUMP TO TEST PSW
020252  100403                           JMPT:   BMI     JMPERR                  ;BR TO ERROR HALT IF N-BIT IS SET
020254  001002                                   BNE     JMPERR                  ;BR TO ERROR HALT IF Z-BIT IS CLEAR
020256  102401                                   BVS     JMPERR                  ;BR TO ERROR HALT IF V-BIT IF SET
020260  103401                                   BCS     TS270
020262                                   JMPERR:
020262  104000                                   EMT                             ;JMP INSTRUCTION AFFECTED CC'S
                                         ;*****************************************************************************
                                         ;
                                         ;       THIS TEST VERIFIES THE SET AND CLEAR CONDITION CODE INSTRUCTIONS.
                                         ;THE TEST CONSISTS OF TWO ROUTINES, ONE TO TEST ALL CLEAR CC
                                         ;INSTRUCTIONS, AND THE SECOND TO TEST ALL SET CC INSTRUCTIONS. ALL
                                         ;POSSIBLE COMBINATIONS OF CONDITION CODES ARE TESTED, INCLUDING NOP'S.
                                         ;       TO TEST THE CLEAR CC INSTRUCTIONS, ALL CONDITION CODES ARE
                                         ;INITIALLY SET. THE INSTRUCTION IS EXECUTED, AND THE PSW IS CHECKED
                                         ;TO VERIFY THE PROPER COMBINATION OF CONDITION CODES.
                                         ;       TO TEST THE SET CC INSTRUCTIONS, THE CONDITION CODES ARE
                                         ;INITIALLY CLEARED, AND ONLY THE REQUIRED BITS ARE SET BY THE SET CC
                                         ;INSTRUCTION. THE CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT
                                         ;ONLY THE REQUIRED BITS WERE SET.
                                         ;
                                         ;*****************************************************************************
                                         ;TEST 270        TEST SET CC AND CLEAR CC INSTRUCTIONS
                                         ;*****************************************************************************
020264                                   TS270:
020264  012767  000240  000024                   MOV     #240,CC3                ;INITIALIZE CLR CC INSTRUCTION CODES
020272  012767  000017  000032                   MOV     #17,CC2                 ;INITIALIZE OCTAL MAP
020300  012767  000261  000074                   MOV     #261,SC3                ;INITIALIZE SET CC INSTRUCTION CODES
020306  012767  000001  000102                   MOV     #1,SC4                  ;INITIALIZE OCTAL MAP
020314  000277                           CLRCD:  SCC                             ;SET ALL CONDITION CODES
020316  000000                           CC3:    0                               ;CONDITION CODE INSTRUCTION
020320  013704  177776                           MOV     @#PS,R4                 ;COPY THE PSW
020324  042704  177760                           BIC     #177760,R4              ;ISOLATE CONDITION CODES
020330  022704                                   CMP     (PC)+,R4                ;CHECK THAT PROPER CC'S WERE CLEARED
020332  000000                           CC2:    0                               ;OCTAL REPRESENTATION OF CC'S
020334  001401                                   BEQ     CON1
020336  104000                                   EMT                             ;CLEAR CC INSTRUCTION FAILED
020340  005367  177766                   CON1:   DEC     CC2                     ;SET NEXT OCTAL MAP OF CC'S
020344  005267  177746                           INC     CC3                     ;GET NEXT CLEAR CC INSTRUCTION
020350  026727  177742  000257                   CMP     CC3,#257                ;TEST FOR CCC INSTRUCTION
020356  003756                                   BLE     CLRCD                   ;GO TEST NEXT INSTRUCTION IF NOT FOUND
020360  026727  177732  000260                   CMP     CC3,#260                ;CHECK FOR NOP=260
020366  001004                                   BNE     SETCD                   ;GO TEST SET CC INSTRUCTIONS
020370  012767  000017  177734                   MOV     #17,CC2                 ;SET OCTAL MAP TO TEST NOP
```

```
5795 020376 000746                            BR    CLRCD          ;GO TEST NOP
5796 020400 000257                    SETCD:  CCC                  ;CLEAR ALL CONDITION CODES
5797 020402 000000                    SC3:    0                    ;CONDITION CODE INSTRUCTION
5798 020404 013704 177776                     MOV   @#PS,R4        ;COY PSW
5799 020410 042704 177760                     BIC   #177760,R4     ;CLEAR AWAY UNWANTED BITS
5800 020414 022704                            CMP   (PC)+,R4       ;CHECK THAT PROPER CC'S WERE SET
5801 020416 000000                    SC4:    0                    ;OCTAL REPRESENTATION OF CC'S
5802 020420 001401                            BEQ   CON2
5803 020422                           CCERR:
5804 020422 104000                            EMT                  ;SET CC FAILED OR SEQUENCE ERROR
5805 020424 005267 177766             CON2:   INC   SC4            ;SET NEXT OCTAL MAP
5806 020430 005267 177746                     INC   SC3            ;PREPARE NEXT SET CC INSTRUCTION
5807 020434 026727 177742 000277              CMP   SC3,#277       ;FINISHED?
5808 020442 003756                            BLE   SETCD          ;BR IF NO
5809 020444 000167 000006                     JMP   MOR0           ;JUMP TO NEXT TESTS
```

```
                                        ;*****************************************************************************
                                        .SBTTL  TEST INSTRUCTIONS USING SAME REGISTER FOR SOURCE & DESTINATION
                                        ;
                                        ;IN AUTO INCREMENT (DECREMENT) MODES AND
                                        ;AUTO INCREMENT (DECREMENT) DEFERRED MODES,
                                        ;CONTENTS OF THE REGISTER IN USED ARE
                                        ;INCREMENTED (DECREMENTED) BY 2
                                        ;BEFORE USED AS THE SOURCE OPERAND.
                                        ;
020450  000000  000000  000000          A:      .WORD 0,0,0
020456                                  MOR0:
                                        ;*****************************************************************************
                                        ;TEST 271        TEST AUTO-INCREMENT MODE, USING R0
                                        ;*****************************************************************************
020456                                  TS271:
020456  005037  020450                          CLR     @#A             ;CLEAR LOC A
020462  012700  020450                          MOV     #A,R0           ;R0 STORES ADDR OF A
020466  060020                                  ADD     R0,(R0)+        ;CHECK THAT R0 IS INCR BY 2 BEFORE
                                                                        ;BEING USED AS THE SOURCE OPERAND
020470  022700  020452                          CMP     #A+2,R0         ;R0 INCR BY 2?
020474  001401                                  BEQ     MOR1
020476  104000                                  EMT                     ;R0 WAS NOT INCREMENTED BY 2
                                                                        ;
020500  022737  020452  020450          MOR1:   CMP     #A+2,@#A        ;CHECK CONTENT OF R0 WAS INCR BY 2 BEFORE
                                                                        ;BEING USED IN THE ''ADD'' INSTR
                                                                        ;LOC A CONTAINS (A+2)?
020506  001401                                  BEQ     TS272
020510  104000                                  EMT                     ;WRONG SUM IN LOC A

                                        ;*****************************************************************************
                                        ;TEST 272        AUTO-DECREMENT MODE, USING R0
                                        ;*****************************************************************************
020512                                  TS272:
020512  005037  020450                          CLR     @#A             ;CLEAR LOC A
020516  012700  020452                          MOV     #A+2,R0         ;R0 STORES ADDR OF A+2
020522  060040                                  ADD     R0,-(R0)        ;CHECK THAT R0 IS DECR BY 2 BEFORE
                                                                        ;BEING USED AS THE SOURCE OPERAND
020524  022700  020450                          CMP     #A,R0           ;R0 DECR BY 2?
020530  001401                                  BEQ     MOR2
020532  104000                                  EMT                     ;R0 WAS NOT DECREMENTED BY 2
                                                                        ;
020534  022737  020450  020450          MOR2:   CMP     #A,@#A          ;CONTENT OF R0 WAS DECR BY 2 BEFORE
                                                                        ;BEING USED IN THE ''ADD'' INSTR
                                                                        ;LOC A CONTAINS (R0)
020542  001401                                  BEQ     TS273
020544  104000                                  EMT                     ;WRONG SUM IN LOC A
                                                                        ;
                                        ;*****************************************************************************
                                        ;TEST 273        TEST AUTO-INCREMENT DEFERRED MODE, USING R0
                                        ;*****************************************************************************
020546                                  TS273:
020546  005037  020450                          CLR     @#A             ;CLEAR LOC A
020552  005037  020454                          CLR     @#A+4           ;CLEAR LOC A+4
020556  012737  020450  020452                  MOV     #A,@#A+2        ;STORE ADDR A IN LOC A+2
020564  012700  020452                          MOV     #A+2,R0         ;R0 STORES ADDR A+2
020570  060030                                  ADD     R0,@(R0)+       ;CHECK THAT R0 IS INCR BY 2 BEFORE
```

```
                                                             ;BEING USED AS THE SOURCE OPERAND
020572  022700  020454                  CMP     #A+4,RO      ;RO INCR BY 2?
020576  001401                          BEQ     MOR3
020600  104000                          EMT                  ;RO WAS NOT INCREMENTED BY 2
                                                             ;
020602  022737  020450  020452  MOR3:   CMP     #A,@#A+2     ;LOC A+2 STILL STORES ADDR A?
020610  001401                          BEQ     MOR4
020612  104000                          EMT                  ;LOC A+2 STORES WRONG DATA
                                                             ;
020614  022737  020454  020450  MOR4:   CMP     #A+4,@#A     ;CHECK CONTENT OF RO WAS INCR BY 2 BEFORE
                                                             ;BEING USED IN THE "ADD" INSTR
020622  001401                          BEQ     MOR5
020624  104000                          EMT                  ;LOC A STORES WRONG DATA
                                                             ;
020626  005737  020454          MOR5:   TST     @#A+4        ;LOC A+4 STILL STORES 0?
020632  001401                          BEQ     TS274
020634  104000                          EMT                  ;LOC A+4 DID NOT STAY CLEAR

                                ;*****************************************************************************
                                ;TEST 274       TEST AUTO-DECREMENT DEFERRED, USING RO
                                ;*****************************************************************************
020636                          TS274:
020636  005037  020450                  CLR     @#A          ;CLEAR LOC A
020642  005037  020454                  CLR     @#A+4        ;CLEAR LOC A+4
020646  012700  020454                  MOV     #A+4,RO      ;RO STORES ADDR A+4
020652  012737  020450  020452          MOV     #A,@#A+2     ;STORE ADDR A IN LOC A+2
020660  060050                          ADD     RO,@-(RO)    ;CHECK THAT RO IS DECR BY 2 BEFORE
                                                             ;BEING USED AS THE SOURCE OPERAND
020662  022700  020452                  CMP     #A+2,RO      ;RO DECREMENTED BY 2?
020666  001401                          BEQ     MOR6
020670  104000                          EMT                  ;RO WAS NOT DECREMENTED BY 2
                                                             ;
020672  022737  020452  020450  MOR6:   CMP     #A+2,@#A     ;CHECK CONTENT OF RO WAS DECR BY 2 BEFORE
                                                             ;BEING USED IN THE "ADD" INSTR
020700  001401                          BEQ     MOR7
020702  104000                          EMT                  ;LOC A STORES WRONG DATA

                                                             ;
020704  022737  020450  020452  MOR7:   CMP     #A,@#A+2     ;LOC A+2 STILL STORES A?
020712  001401                          BEQ     MOR8
020714  104000                          EMT                  ;LOC A+2 STORES WRONG DATA
                                                             ;
020716  005737  020454          MOR8:   TST     @#A+4        ;LOC A+4 STILL STORES 0?
020722  001401                          BEQ     TS275
020724  104000                          EMT                  ;LOC A+4 DID NOT STAY CLEAR
                                                             ;
```

```
                                      ;*****************************************************************************
                                      .SBTTL  INSTRUCTION USING PC AS SOURCE REGISTER
                                      ;
                                      ;IN INDEX, INDEX DEFERRED, RELATIVE, AND
                                      ;RELATIVE DEFERRED MODES, DESTINATION WILL CONTAIN
                                      ;THE PC COUNT OF THE CURRENT INSTRUCTION +4.
                                      ;
                                      ;*****************************************************************************
                                      ;TEST 275        TEST PC AS SOURCE IN MODE 0, USING R0
                                      ;*****************************************************************************
020726                                TS275:
020726  012700  177777                        MOV     #-1,R0          ;SET ALL 1 IN R0
020732  010700                        PCN01:  MOV     PC,R0           ;STORES PC IN R0
020734  022700  020734                        CMP     #PCN01+2,R0     ;R0 STORES PC+2?
020740  001401                                BEQ     TS276
020742  104000                                EMT                     ;R0 STORED WRONG VALUE

                                      ;*****************************************************************************
                                      ;TEST 276        TEST PC AS SOURCE IN MODE 6, USING R0
                                      ;*****************************************************************************
020744                                TS276:
020744  012700  020450                        MOV #A,R0               ;R0 STORES ADDR A
020750  010760  000004                PCN2:   MOV PC,4(R0)            ;EFFECTIVE ADDR IS A+4
020754  022737  020754  020454                CMP     #PCN2+4,@#A+4   ;LOC A+4 STORES PC+4?
020762  001401                                BEQ     TS277
020764  104000                                EMT                     ;LOC A+4 STORED WRONG VALUE

                                      ;*****************************************************************************
                                      ;TEST 277        TEST PC AS SOURCE IN MODE 7, USING R0
                                      ;*****************************************************************************
020766                                TS277:
020766  012737  020450  020454                MOV     #A,@#A+4        ;LOC A+4 STORES ADDR A
020774  012700  020450                        MOV     #A,R0           ;R0 STORES ADDR A
021000  010770  000004                PCN3:   MOV     PC,@4(R0)       ;EFFECTIVE ADDR IS A
021004  022737  021004  020450                CMP     #PCN3+4,@#A     ;LOC A STORES PC+4?
021012  001401                                BEQ     TS300
021014  104000                                EMT                     ;LOC A STORED WRONG VALUE

                                      ;*****************************************************************************
                                      ;TEST 300        TEST PC AS SOURCE IN RELATIVE DEFERRED MODE ,USING R0
                                      ;*****************************************************************************
021016                                TS300:
021016  012737  020452  020450                MOV     #A+2,@#A        ;LOC A STORES ADDR A+2
021024  010777  177420                PCN4:   MOV     PC,@A           ;EFFECTIVE ADDR IS A+2
021030  022737  021030  020452                CMP     #PCN4+4,@#A+2   ;LOC A+2 STORES PC+4?
021036  001401                                BEQ     TS301
021040  104000                                EMT                     ;LOC A+2 STORED WRONG VALUE

                                      ;*****************************************************************************
                                      ;TEST 301        TEST PC AS SOURCE IN RELATIVE MODE ,USING R0
                                      ;*****************************************************************************
021042                                TS301:
021042  005037  020450                        CLR     @#A             ;CLEAR A
021046  010767  177376                PCN5:   MOV     PC,A            ;EFFECTIVE ADDR IS A
021052  022737  021052  020450                CMP     #PCN5+4,@#A     ;LOC A STORES PC+4?
021060  001401                                BEQ     TS302
```

```
021062  104000                          EMT                             ;LOCATION A STORED WRONG VALUE
                                ;***********************************************************************
                                ;THIS TESTS THE MOVE FROM PROCESSOR TYPE INSTRUCTION(MFPT)
                                ;UPON EXECUTION R0 WILL RECIEVE THE PROCESSOR MODEL CODE
                                ;WHICH IS "000003" FOR THE DCF11-AA
                                ;***********************************************************************
                                ;***********************************************************************
                                ;TEST 302       TEST MFPT
                                ;***********************************************************************
021064                          TS302:
        000007                  MFPT=000007
021064  012706  001000                  MOV     #STBOT,SP       ;INITIALIZE STACK POINT IN CASE OF TRAP
021070  000007                          MFPT                    ;GET MODEL CODE.IF THIS TRAPS AN ERROR WILL BE REPORTED
021072  022700  000003                  CMP     #3,R0           ;CHECK IF CORRECT CODE RETURNED
021076  001401                          BEQ     TS303
021100  104000                          EMT                     ;WRONG CODE RETURNED



                                ;***********************************************************************
                                .SBTTL  THE NEXT THREE TESTS EXERCISE MASKING ACTION OF MICROCODES.
                                ;***********************************************************************
                                ;TEST 303       TEST SUB INSTRUCTION, SM=0, DM=2
                                ;***********************************************************************
021102                          TS303:
021102  012737  052525  000000          MOV     #052525,@#0     ;SET UP LOC 0
021110  012701  050505                  MOV     #050505,R1      ;SET UP R1
021114  005000                          CLR     R0              ;CLEAR R0
021116  160120                          SUB     R1,(R0)+        ;SUBTRACTION, SM=0,DM=2
021120  022737  002020  000000          CMP     #2020,@#0       ;CHECK DIFFERENCE AT LOC 0
021126  001401                          BEQ     TS304
021130  104000                          EMT                     ;WRONG RESULT FROM SUBTRACTION

                                ;***********************************************************************
                                ;TEST 304       TEST MFPD WITH R0, IN MODE 2
                                ;***********************************************************************
021132                          TS304:
021132  012737  052525  000000          MOV     #052525,@#0     ;SET UP LOC 0
021140  005000                          CLR     R0              ;CLEAR R0
021142  012767  170000  156626          MOV     #170000,PS      ;SET USER MODE ON, CURRENT & PREVIOUS
021150  012706  000600                  MOV     #USESTK,R6      ;SET USER STACK POINTER
021154  106520                          MFPD    (R0)+           ;MODE 2, MFPD
021156  005067  156614                  CLR     PS              ;SET KERNEL MODE
021162  022767  052525  157406          CMP     #052525,USESTK-2        ;CHECK DATA ON STACK
021170  001401                          BEQ     TS305
021172  104000                          EMT                     ;INCORRECT DATA FROM MFPD

                                ;***********************************************************************
                                ;TEST 305       TEST MTPD WITH R0, IN MODE 2
                                ;***********************************************************************
021174                          TS305:
021174  012767  170000  156574          MOV     #170000,PS      ;SET USER MODE ON, CURRENT & PREVIOUS
021202  012706  000600                  MOV     #USESTK,R6      ;SET USER STACK POINTER
021206  012746  125252                  MOV     #125252,-(R6)   ;PUSH DATA IN USER STACK
021212  012737  000000  000000          MOV     #0,@#0          ;CLEAR LOC 0
021220  005000                          CLR     R0              ;CLEAR R0
```

```
021222  106620                          MTPD    (R0)+                   ;MODE 2, MTPD
021224  005067  156546                  CLR     PS                      ;SET KERNEL MODE
021230  022737  125252  000000          CMP     #125252,@#0             ;CHECK DATA ON LOC 0
021236  001463                          BEQ     TESTN1
021240  104000                          EMT                             ;INCORRECT DATA FROM MTPD

021242  000402                  BRTAB:  BR      .+6
021244  001002                          BNE     .+6
021246  001402                          BEQ     .+6
021250  002002                          BGE     .+6
021252  002402                          BLT     .+6
021254  003002                          BGT     .+6
021256  003402                          BLE     .+6
021260  100002                          BPL     .+6
021262  100402                          BMI     .+6
021264  101002                          BHI     .+6
021266  101402                          BLOS    .+6
021270  102002                          BVC     .+6
021272  102402                          BVS     .+6
021274  103002                          BCC     .+6                     ;SAME AS BHIS
021276  103402                          BCS     .+6                     ;SAME AS BLO

        000002                  .RADIX  2
021300  177777                  YNTAB:  1111111111111111                ;BR
021302  170360                          1111000011110000                ;BNE:  Z=0
021304  007417                          0000111100001111                ;BEQ:  Z=1
021306  146063                          1100110000110011                ;BGE:  N XOR V =0
021310  031714                          0011001111001100                ;BLT:  N XOR V =1
021312  140060                          1100000000110000                ;BGT:  Z+(N XOR V) =0
021314  037717                          0011111111001111                ;BLE:  Z+(N XOR V) =1

021316  177400                          1111111100000000                ;BPL:  N=0
021320  000377                          0000000011111111                ;BMI:  N=1
021322  120240                          1010000010100000                ;BHI:  C+Z=0
021324  057537                          0101111101011111                ;BLOS: C+Z=1
021326  146314                          1100110011001100                ;BVC:  V=0
021330  031463                          0011001100110011                ;BVS:  V=1
021332  125252                          1010101010101010                ;BCC:  C=0
021334  052525                          0101010101010101                ;BCS:  C=1
        000010                  .RADIX  8

                                ;*******************************************************************************
                                ; THE FOLLOWING ARE SPECIAL CPU TRAP
                                ;HANDLERS TO TRAP AND REPORT SPECIAL TRAPS.
                                ;
                                ;*******************************************************************************

021336                          T04:
021336  104000                          EMT                             ;TRAPPED THRU LOC. 4
021340                          T010:
021340  104000                          EMT                             ;TRAPPED THRU LOC. 10
021342                          T014:
021342  104000                          EMT                             ;TRAPPED THRU LOC. 14
021344                          T020:
021344  104000                          EMT                             ;TRAPPED THRU LOC. 20
021346                          T030:
```

```
021346  104000                            EMT                              ;TRAPPED THRU LOC. 30
021350                           T034:
021350  104000                            EMT                              ;TRAPPED THRU LOC. 34
021352                           T0114:
021352  104000                            EMT                              ;TRAPPED THRU LOC. 114
021354                           T0244:
021354  104000                            EMT                              ;TRAPPED THRU LOC. 244
021356                           T0250:
021356  104000                            EMT                              ;TRAPPED THRU LOC. 250
                                 .SBTTL ** STARTING OF TRAP TEST **

                                 ;SPECIAL CASE OF ODD;.EVEN .BYTE AND REGISTER 6
        000000                   HERE=0

021360  000000                   K1:      0
021362  000000                   K2:      0
021364  000000                   K3:      0
021366  000000                   K4:      0
021370  000000                   K5:      0
021372  000000                   K6:      0
021374  052525                   K7:      052525
021376  052400                   K10:     052400
021400  000000                   K11:     0
021402  000000                   K12:     0
021404  000176                   SWR:     176

021406  032737  000001  001020   TESTN1: BIT      #1,@#$ENV
021414  001403                            BEQ      1$
021416  012767  001022  177760            MOV      #$SWREG,SWR
021424                           1$:
                                 ;*****************************************************************************
                                 ;TEST 306         TEST AUTO INCREMENT AND DECREMENT OF R6 FOR WORD AND BYTES
                                 ;*****************************************************************************
021424                           TS306:
021424  005006                            CLR      %6
021426  112667  156346                    MOVB     (6)+,HERE        ;SIX SHOULD INCREMENT BY TWO
021432  020627  000002                    CMP      %6,#2
021436  001401                            BEQ      BR1
021440  104000                            EMT                       ;R6 DID NOT AUTO INCREMENT BY TWO

021442  012706  001000           BR1:     MOV      #1000,%6
021446  114627  000000                    MOVB     -(6),#HERE       ;SHOULD DECREMENT BY TWO
021452  020627  000776                    CMP      %6,#776
021456  001401                            BEQ      BR2
021460  104000                            EMT                       ;R6 DID NOT AUTO DECREMENT BY 2

021462  005006                   BR2:     CLR      %6
021464  112626                            MOVB     (6)+,(6)+        ;DOUBLES AUTO INCREMENT OF R6
021466  020627  000004                    CMP      %6,#4
021472  001401                            BEQ      BR3
021474  104000                            EMT                       ;WRONG AUTO INCREMENT OF R6

021476  005006                   BR3:     CLR      %6
021500  005004                            CLR      %4
021502  122624                            CMPB     (6)+,(4)+        ;TEST INCREMENT OF R6
021504  020627  000002                    CMP      %6,#2
```

```
021510  001401                            BEQ     BR4
021512  104000                            EMT                     ;WRONG INCREMENT OF R6

021514  005006                    BR4:    CLR     %6
021516  005004                            CLR     %4
021520  122426                            CMPB    (4)+,(6)+       ;TEST INCREMENT OF R6
021522  020627  000002                    CMP     %6,#2
021526  001401                            BEQ     BR5
021530  104000                            EMT                     ;WRONG INCREMENT OF R6

021532  005006                    BR5:    CLR     %6
021534  005004                            CLR     %4
021536  122624                            CMPB    (6)+,(4)+       ;TEST INCREMENT OF R4
021540  020427  000001                    CMP     %4,#1
021544  001401                            BEQ     BR6
021546  104000                            EMT                     ;WRONG INCREMENT OF R4
021550  005006                    BR6:    CLR     %6
021552  005004                            CLR     %4
021554  122426                            CMPB    (4)+,(6)+       ;TEST INCREMENT OF R6
021556  020627  000002                    CMP     %6,#2
021562  001401                            BEQ     BR7
021564  104000                            EMT                     ;WRONG INCREMENT OF R6

021566  005006                    BR7:    CLR     %6
021570  005004                            CLR     %4
021572  122426                            CMPB    (4)+,(6)+       ;TEST INCREMENT OF R4
021574  020427  000001                    CMP     %4,#1
021600  001401                            BEQ     BR10
021602  104000                            EMT                     ;WRONG INCREMENT OF R4

021604  012706  001000            BR10:   MOV     #1000,%6
021610  124627  000000                    CMPB    -(6),#HERE      ;TEST DECREMENT OF R6
021614  022706  000776                    CMP     #776,%6
021620  001401                            BEQ     TS307           .
021622  104000                            EMT                     ;WRONG DECREMENT OF R6,OR WRONG $TSTNM
                                  ;*************************************************************************
                                  ;TEST 307        TEST TRANSFER OF .BYTE USING R6
                                  ;*************************************************************************
021624                            TS307:
021624. 012767  123456  177536            MOV     #123456,K5
021632  012767  050505  177520            MOV     #050505,K1
021640  012705  021360                    MOV     #K1,%5          ;%5=(050505)K1
021644  012706  021370                    MOV     #K5,%6          ;%6=(123456)K5
021650  112625                            MOVB    (6)+,(5)+       ;LOW .BYTE OF R6 TO R5
021652  022767  050456  177500            CMP     #050456,K1
021660  001401                            BEQ     BR11
021662  104000                            EMT                     ;FALSE TRANSFER OF .BYTE

021664  012767  123456  177476    BR11:   MOV     #123456,K5
021672  012767  050505  177460            MOV     #050505,K1
021700  012705  021360                    MOV     #K1,%5          ;%5(050505)K1
021704  012706  021372                    MOV     #K6,%6          ;%6(123456)K5
021710  114625                            MOVB    -(6),(5)+       ;LOW .BYTE OF R6 TO R5 (DECREMENT)
021712  026727  177442  050456            CMP     K1,#050456
021720  001401                            BEQ     BR12
021722  104000                            EMT                     ;FALSE R6 .BYTE TRANSFER
```

```
021724  012767  123456  177426  BR12:   MOV     #123456,K1
021732  012767  050505  177430          MOV     #050505,K5
021740  012705  021360                  MOV     #K1,%5          ;(123456)
021744  012706  021370                  MOV     #K5,%6          ;(050505)
021750  112526                          MOVB    (5)+,(6)+       ;LOW OF R5 TO LOW OF R6
021752  022767  050456  177410          CMP     #050456,K5
021760  001401                          BEQ     BR13
021762  104000                          EMT                     ;FALSE R6 .BYTE TRANSFER

021764  012767  123456  177366  BR13:   MOV     #123456,K1
021772  012767  050505  177370          MOV     #050505,K5
022000  012705  021361                  MOV     #K1+1,%5        ;123456
022004  012706  021370                  MOV     #K5,%6          ;050505
022010  112526                          MOVB    (5)+,(6)+       ;HIGH OF R5 TO LOW OF R6
022012  026727  177352  050647          CMP     K5,#050647
022020  001401                          BEQ     BR14
022022  104000                          EMT                     ;FALSE R6 .BYTE TRANSFER

022024  012767  123456  177326  BR14:   MOV     #123456,K1
022032  012767  050505  177330          MOV     #050505,K5
022040  012705  021361                  MOV     #K1+1,%5        ;R5-123456-ODD ADDRESS
022044  012706  021370                  MOV     #K5,%6          ;R6-050505--.EVEN ADDRESS
022050  112625                          MOVB    (6)+,(5)+       ;LOW OF R6 TO HIGH OF R5
022052  022767  042456  177300          CMP     #042456,K1
022060  001401                          BEQ     TS310
022062  104000                          EMT                     ;FAILED LOW OF 6 TO HIGH OF 5,OR WRONG $TSTNM
                                ;*****************************************************************************
                                ;TEST 310       TEST BYTE OPERATION WITH SEQUENTIAL ODD-EVEN ADDRESS
                                ;*****************************************************************************
022064                          TS310:
022064  126767  177304  177303          CMPB    K7,K7+1         ;SAME .WORD LOW TO HIGH
022072  001401                          BEQ     BR15
022074  104000                          EMT                     ;SHOULD COMPARE LOW TO HIGH

022076  126767  177273  177270  BR15:   CMPB    K7+1,K7         ;COMPARE ODD TO .EVEN SAME .WORD
022104  001401                          BEQ     BR16
022106  104000                          EMT                     ;ODD TO .EVEN .BYTE FAILURE

022110  126767  177263  177256  BR16:   CMPB    K10+1,K7        ;SEQUENTIAL .BYTES
022116  001401                          BEQ     BR17
022120  104000                          EMT                     ;ODD TO .EVEN FAILED

022122  126767  177250  177242  BR17:   CMPB    K10,K6
022130  001401                          BEQ     BR20
022132  104000                          EMT                     ;.EVEN TO EVEN FAILED
022134  126767  177235  177235  BR20:   CMPB    K7+1,K10+1
022142  001401                          BEQ     BR21
022144  104000                          EMT                     ;ODD TO ODD FAILED

022146  126767  177224  177223  BR21:   CMPB    K10,K10+1
022154  001001                          BNE     BR22
022156  104000                          EMT                     ;LOW TO HIGH IN SAME .WORD FAILED

022160  126767  177213  177211  BR22:   CMPB    K10+1,K10+1
022166  001401                          BEQ     BR23
```

```
022170  104000                           EMT                          ;HIGH TO LOW IN SAME .WORD FAILED

022172  126767  177200  177175  BR23:    CMPB    K10,K7+1
022200  001001                           BNE     TS311
022202  104000                           EMT                          ;.EVEN TO ODD FAILED,OR WRONG $TSTNM


                                ;*************************************************************************
                                ;TEST 311        TEST THAT DECREMENT R6 TO A VALUE LESS THAN 400 TRAPS
                                ;*************************************************************************
022204                          TS311:
022204  012706  000150                   MOV     #150,%6              ;R6 = 150
022210  012767  022222  155566           MOV     #TDEC1,4             ;STACK OVERFLOW TRAP POINTER
022216  005746                           TST     -(6)                 ;WITH R6 = 150 SHOULD TRAP
022220  104000                           EMT                          ;SHOULD HAVE TRAPPED,OR WRONG $TSTNM
022222                          TDEC1:

                                ;*************************************************************************
                                ;TEST 312        TEST FOR DECREMENT OF R6 ON OVERFLOW TRAP
                                ;*************************************************************************
022222                          TS312:
022222  012706  000150                   MOV     #150,%6              ;R6 = 150
022226  012767  022236  155550           MOV     #TDEC2,4             ;TRAP POINTER
022234  005746                           TST     -(6)                 ;WITH R6 = 150 SHOULD TRAP
022236  020627  000142          TDEC2:   CMP     %6,#142              ;DID R6 DECREMENT
022242  001401                           BEQ     TS313
022244  104000                           EMT                          ;R6 NOT = 142,OR WRONG $TSTNM

                                ;*************************************************************************
                                ;TEST 313        TEST DIFFERENT TYPES OF OVERFLOW
                                ;*************************************************************************
022246                          TS313:
022246  012706  000150                   MOV     #150,%6
022252  005067  155670                   CLR     146                  ;STATUS WORD OF LOC 10
022256  012767  022266  155520           MOV     #TDEC3,4             ;RETURN TO LOC 4
022264  005246                           INC     -(6)
022266  005767  155654          TDEC3:   TST     146
022272  001001                           BNE     1$
022274  104000                           EMT                          ;INCREMENT OPERATION NOT INHIBITED
022276  012705  001000          1$:      MOV     #1000,%5
022302  012706  000400                   MOV     #400,%6
022306  012767  022320  155470           MOV     #TDEC4,4
022314  124645                           CMPB    -(6),-(5)
022316  104000                           EMT                          ;STACK = 400 AND DECREMENTED, SHOULD TRAP
022320  012706  000400          TDEC4:   MOV     #400,%6
022324  012767  022336  155452           MOV     #TDEC7,4
022332  134546                           BITB    -(5),-(6)
022334                          TDEC6:
022334  104000                           EMT                          ;NO STACK OVERFLOW,OR WRONG $TSTNM
022336                          TDEC7:

                                ;*************************************************************************
                                ;TEST 314        TEST THAT AN 77 CAUSES AN OVERFLOW TRAP
                                ;*************************************************************************
022336                          TS314:
022336  012706  000400                   MOV     #400,%6              ;SET UP STACK TO OVERFLOW
```

```
022342 012767 022360 155440           MOV     #VDEC2,10          ;SET UP 77 VECTOR
022350 012767 022364 155426           MOV     #VDEC,4 ;SET UP OVERFLOW VECTOR
022356 000077                         77                         ;THIS TRAP SHOULD CAUSE OVERFLOW
022360 000167 157544          VDEC2:  JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022364 012767 021340 155416   VDEC:   MOV     #T010,10           ;RESTORE VECTOR
                              ;*************************************************************************
                              ;TEST 315         TEST THAT AN IOT CAUSES AN OVERFLOW TRAP
                              ;*************************************************************************
022372                        TS315:
022372 012706 000400                  MOV     #400,%6            ;SET UP STACK TO OVERFLOW
022376 012767 022414 155414           MOV     #VDEC4,20          ;SET UP IOT VECTOR
022404 012767 022420 155372           MOV     #VDEC3,4           ;SET UP OVERFLOW VECTOR
022412 000004                         IOT                        ;THIS TRAP SHOULD CAUSE OVERFLOW
022414 000167 157510          VDEC4:  JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022420 012767 021344 155372   VDEC3:  MOV     #T020,20           ;RESTORE VECTOR

                              ;*************************************************************************
                              ;TEST 316         TEST THAT AN EMT CAUSES AN OVERFLOW TRAP (CHECK OF YELLOW ZONE)
                              ;*************************************************************************
022426                        TS316:
022426 012706 000400                  MOV     #400,%6            ;SET UP STACK TO OVERFLOW
022432 012767 022450 155370           MOV     #VDEC6,30          ;SET UP INST VECTOR
022440 012767 022454 155336           MOV     #VDEC5,4           ;SET UP OVERFLOW BECTOR
022446 104000                         EMT                        ;THIS TRAP SHOULD CAUSE OVERFLOW
022450 000167 157454          VDEC6:  JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022454 012767 002130 155346   VDEC5:  MOV     #ERROR1,30         ;RESTORE VECTOR

                              ;*************************************************************************
                              ;TEST 317         TEST THAT AN TRAP CAUSES AN OVERFLOW TRAP
                              ;*************************************************************************
022462                        TS317:
022462 012706 000400                  MOV     #400,%6            ;SET UP STACK TO OVERFLOW
022466 012767 022504 155340           MOV     #VDEC8,34          ;SET UP TRAP VECTOR
022474 012767 022510 155302           MOV     #VDEC7,4           ;SET UP OVERFLOW VECTOR
022502 104400                         TRAP                       ;THIS TRAP SHOULD CAUSE OVERFLOW
022504 000167 157420          VDEC8:  JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022510 012767 021350 155316   VDEC7:  MOV     #T034,34           ;RESTORE VECTOR
                              ;*************************************************************************
                              ;TEST 320         TEST THAT AN TRT CAUSES AN OVERFLOW TRAP
                              ;*************************************************************************
022516                        TS320:
022516 012706 000400                  MOV     #400,%6            ;SET UP STACK TO OVERFLOW
022522 012767 022540 155264           MOV     #VDEC10,14         ;SET UP TRT VECTOR
022530 012767 022544 155246           MOV     #VDEC9,4           ;SET UP OVERFLOW VECTOR
022536 000003                         TRT                        ;THIS TRAP SHOULD CAUSE OVERFLOW
022540 000167 157364          VDEC10: JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022544 012767 021342 155242   VDEC9:  MOV     #T014,14           ;RESTORE VECTOR
                              ;*************************************************************************
                              ;TEST 321         TEST THAT AN ILLA CAUSES AN OVERFLOW TRAP
                              ;*************************************************************************
022552                        TS321:
022552 012706 000400                  MOV     #400,%6            ;SET UP STACK TO OVERFLOW
022556 012767 022574 155220           MOV     #VDEC11,4          ;SET UP ILLA VECTOR
022564 012767 022600 155212           MOV     #VDEC12,4          ;SET UP OVERFLOW VECTOR
022572 004700                         ILLA                       ;THIS TRAP SHOULD CAUSE OVERFLOW
022574 000167 157330          VDEC11: JMP     ERROR1             ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
```

```
022600 012767 021336 155176 VDEC12: MOV   #T04,4   ;RESTORE VECTOR
022606 020627 000370                CMP   %6,#370          ;STACK PUSHED FOUR WORDS?
022612 001401                       BEQ   TS322
022614 104000                       EMT                    ;TRAP OVERFLOW DID NOT OCCUR
                                    ;*************************************************************************
                                    ;TEST 322       TEST THAT AN ILLB CAUSES AN OVERFLOW TRAP
                                    ;*************************************************************************
022616                              TS322:
022616 012706 000400                MOV   #400,%6          ;SET UP STACK TO OVERFLOW
022622 012767 022640 155154         MOV   #VDEC13,4        ;SET UP ILLB VECTOR
022630 012767 022644 155146         MOV   #VDEC14,4        ;SET UP OVERFLOW VECTOR
022636 000100                       ILLB                   ;THIS TRAP SHOULD CAUSE OVERFLOW
022640 000167 157264         VDEC13: JMP  ERROR1           ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
022644 012767 021336 155132  VDEC14: MOV  #T04,4   ;RESTORE VECTOR

                                    ;*************************************************************************
                                    ;TEST 323       TEST FOR FALSE OVERFLOW TRAP
                                    ;*************************************************************************
022652                              TS323:

022652 012767 022720 155124         MOV   #FOVER,4         ;SET UP OVERFLOW POINTER
022660 012706 001002                MOV   #1002,%6
022664 005746                       TST   -(6)             ;SHOULD NOT OVERFLOW
022666 012706 002002                MOV   #2002,%6
022672 005746                       TST   -(6)             ;SHOULD NOT OVERFLOW
022674 012706 004002                MOV   #4002,%6
022700 005746                       TST   -(6)             ;SHOULD NOT OVERFLOW
022702 012706 010002                MOV   #10002,%6
022706 005746                       TST   -(6)
022710 012706 020000                MOV   #20000,%6        ;SHOULD NOT OVERFLOW
022714 005746                       TST   -(6)
022716 000401                       BR    STP
022720                              FOVER:
022720 104000                       EMT                    ;IT OVERFLOWED,OR WRONG $TSTNM
022722 012767 021336 155054  STP:   MOV   #T04,4
022730 005067 155052                CLR   6
                                    ;*************************************************************************
                                    ;TEST 324       TEST THAT BIT 4 PSW WILL CAUSE A TRAP TO 14
                                    ;*************************************************************************
022734                              TS324:
022734 012706 001000                MOV   #STBOT,SP
022740 012767 022764 155046         MOV   #RETAT,RTRAP4    ;SET UP TO TRAP TO 14
022746 012746 000020                MOV   #20,-(SP)        ;PUSH T BIT
022752 012746 022760                MOV   #.+6,-(SP)       ;PUSH PC
022756 000002                       RTI                    ;SET T BIT
022760 000240                       NOP                    ;TRAP HERE
022762 104000                       EMT                    ;TRACE BIT DID NOT TRAP!,OR WRONG $TESTN
022764                              RETAT:
                                    ;*************************************************************************
                                    ;TEST 325       TEST  STACK POINTER DECREMENTS
                                    ;*************************************************************************
022764                              TS325:
022764 012706 001000                MOV   #STBOT,SP
022770 012767 023014 155016         MOV   #RETBT,RTRAP4
022776 012746 000020                MOV   #20,-(SP)        ;PUSH T BIT
023002 012746 023010                MOV   #.+6,-(SP)       ;PUSH PC
```

```
023006  000002                           RTI                      ;SET T BIT
023010  000240                           NOP                      ;TRAP HERE
023012  104000                           EMT                      ;TRACE BIT DID NOT TRAP!
023014  020627  000774          RETBT:   CMP     SP,#STBOT-4
023020  001401                           BEQ     TS326
023022  104000                           EMT                      ;STACK POINTER WAS NOT PUSHED BY TRAP,OR WRONG $TESTN
                                ;*****************************************************************************
                                ;TEST 326          TEST  FOR PROPER PC ON STACK
                                ;*****************************************************************************
023024                          TS326:
023024  012706  001000                   MOV     #STBOT,SP
023030  012767  023050  154756           MOV     #RETCT,RTRAP4
023036  012746  000020                   MOV     #20,-(SP)        ;PUSH T BIT
023042  012746  023050                   MOV     #.+6,-(SP)       ;PUSH PC
023046  000002                           RTI                      ;SET T BIT
                                                                  ;TRAP HERE
023050  022767  023050  155716  RETCT:   CMP     #.,STBOT-4
023056  001401                           BEQ     TS327
023060  104000                           EMT                      ;CORRECT PC WAS NOT SAVED ON STACK,OR WRONG $TESTN



                                ;*****************************************************************************
                                ;TEST 327          TEST  THAT RTT POPS T- BIT
                                ;*****************************************************************************
023062                          TS327:

023062  012706  001000                   MOV     #STBOT,SP
023066  005001                           CLR     R1               ;CLEAR R1
023070  012746  000020                   MOV     #20,-(SP)
023074  012746  023110                   MOV     #RTT1,-(SP)
023100  012767  023116  154706           MOV     #RTT2,14
023106  000006                           RTT
023110  000240                  RTT1:    NOP
023112  001401                           BEQ     TS330
023114  104000                           EMT                      ;T-BIT DID NOT TRAP,OR WRONG $TESTN

023116                          RTT2:
                                ;*****************************************************************************
                                ;TEST 330          TEST  THAT RTT ALLOWS ONE INST. BEFORE TRAP
                                ;*****************************************************************************
023116                          TS330:
023116  012705  177777                   MOV     #177777,%5
023122  012706  001000          RTT5:    MOV     #STBOT,SP
023126  012746  000020                   MOV     #20,-(SP)
023132  012746  023150                   MOV     #RTT3,-(SP)
023136  012767  023160  154650           MOV     #RTT4,14
023144  005001                           CLR     R1               ;CLEAR R0
023146  000006                           RTT                      ;SET T-BIT
023150  005201                  RTT3:    INC     R1
023152  005205                           INC     %5
023154  001762                           BEQ     RTT5             ;DO THIS TEST NO MORE THAN 2 TIMES
023156  104000                           EMT                      ;DID NOT TRAP
023160  005301                  RTT4:    DEC     R1               ;SEE IF RTT ALLOWS 1 INST.
023162  001403                           BEQ     RTT6
023164  005205                           INC     %5               ;DO THIS TEST NO MORE THAN TWO TIMES
023166  001755                           BEQ     RTT5
```

```
023170  104000                          EMT                             ;RTT DID NOT ALLOW 1 INST.,OR WRONG $TESTN
023172                          RTT6:
                                ;*******************************************************************************
                                ;TEST 331               TEST  THAT RTI DOES NOT ALLOW 1 INST.
                                ;*******************************************************************************
023172                          TS331:
023172  012706  001000                  MOV     #STBOT,SP
023176  012746  000020                  MOV     #20,-(SP)
023202  012746  023220                  MOV     #RTI1,-(SP)
023206  012767  023224  154600          MOV     #RTI2,14
023214  005001                          CLR     R1
023216  000002                          RTI                             ;SET T-BIT
023220  005201                  RTI1:   INC     R1                      ;RTI SHOULD NOT ALLOW THIS
023222  104000                          EMT                             ;T- BIT DID NOT CAUSE TRAP
023224  005701                  RTI2:   TST     R1
                                                                        ;RTI SHOULD NOT ALLOW 1 INST. BEFORE TRAP
023226  001401                          BEQ     TS332
023230  104000                          EMT                             ;RTI DID ALLOW 1 INST. BEFORE TRAP,OR WRONG $TESTN

                                ;*******************************************************************************
                                ;TEST 332               TEST TRAP ON TRAP THAT TRACE BIT TRAPS ARE INHIBITED ON TRAP INST
                                ;*******************************************************************************
023232                          TS332:

023232  012706  001000                  MOV     #STBOT,%6
023236  012767  023276  154550          MOV     #TRACE,14               ;TRACE TRAP
023244  005027  000016                  CLR     #16
023250  005027  000022                  CLR     #22
023254  012767  023302  154536          MOV     #TONT1,20               ;IOT TRAP
023262  012746  000020                  MOV     #20,-(SP)               ;PUSH T BIT
023266  012746  023274                  MOV     #.+6,-(SP)              ;PUSH PC
023272  000006                          RTT
023274  000004                          IOT                             ;TRAP, NEW CC HAVE TRACE RESET
023276                          TRACE:
023276  104000                          EMT                             ;TRACE TRAP WAS NOT INHIBITED
023300                          BR70:
023300  104000                          EMT                             ;WRONG TSTNM,OR WRONG $TSTNM
023302  012767  000016  154504  TONT1:  MOV     #16,14
023310  012767  000022  154502          MOV     #22,20
```

```
                                            ;*******************************************************************************
                                            ;TEST 333       TEST THAT THE TRACE BIT IS SAVED IN THE STACK
                                            ;*******************************************************************************
 023316                                     TS333:
 023316  012706  001000                             MOV     #STBOT,%6               ;SET UP STACK POINTER
 023322  012767  023346  154464                     MOV     #TRC1,14                ;TRACE TRAP RETURN
 023330  005067  154462                             CLR     16
 023334  012746  000020                             MOV     #20,-(SP)               ;SET THE T BIT
 023340  012746  023346                             MOV     #TRC1,-(SP)
 023344  000002                                     RTI
 023346  036727  155424  000020     TRC1:           BIT     STBOT-2,#20             ;CHECK FOR T BIT ON STACK
 023354  001001                                     BNE     STP3D
 023356                                     STP3:
 023356  104000                                     EMT                             ;T BIT NOT SAVED ON THE STACK,OR WRONG $TSTNM
 023360  012767  021342  154426     STP3D:          MOV     #T014,14
```

```
                                          ;*******************************************************************
                                          ;THIS ROUTINE TESTS THAT NO LEGAL ADDRESS TRAPS AND THAT AN ILLEGAL
                                          ;ADDRESS TRAPS TO LOCATION 4.  THIS WILL RUN ON 30K SYSTEM.  BUT IF
                                          ;SWITCH REGISTER BIT 1=0, THEN THE MEMORY FROM 28K-30K IS NOT LOOKED
                                          ;AT, SINCE IT MAY HAVE I/O DEVICES.  IF SWR BIT 1=1, THEN THAT AREA IS
                                          ;CHECKED.  (IT SHOULD EITHER ALL TRAP OR ALL NOT TRAP).  LOC 160000
                                          ;IS NO LONGER GUARANTEED TO TRAP, SINCE IT MAY CONTAIN MEMORY. LOCATION
                                          ;177700 (THE UNIBUS ADDRESS FOR R0 ON OLDER SYSTEMS) IS USED FOR FORCING
                                          ;A TIMEOUT IN THE EVENT THAT THERE WAS NO TIMEOUT FROM 0K-28K OR 30K.
                                          ;THIS ROUTINE TESTS MEMORY UNTIL IT DOES A NXM STOP
                                          ;*************************************************************************
                                          ;TEST 334        TEST NON-EXISTENT ADDRESS TRAPS
                                          ;*************************************************************************
023366                                    TS334:
023366  005000                            1$:     CLR     R0                      ;
023370  005067  154412                            CLR     6                       ;
023374  012767  023460  154402                    MOV     #ATRAP,4                ;SET UP ADDRESS TRAP ENTRANCE
023402  012706  001000                            MOV     #STBOT,SP               ;SET STACK POINTER
023406  105720                            NOR:    TSTB    (0)+                    ;IF OUTSIDE OF CORE, TRAP TO 4
023410  020027                                    CMP     R0,(PC)+                ;IS POINTER INSIDE 28K (30K) CORE
023412  160000                            HICORE: .WORD   160000                  ;MAY BE CHANGED TO 170000 IF 30K
023414  103774                                    BLO     NOR                     ;TEST THE REST OF CORE
023416  012737  023432  000004                    MOV     #R0TRAP,@#4             ;SET UP NEW VECTOR POINTER
023424  105737  177700                            TSTB    @#177700                ;SHOULD CAUSE A TRAP
023430                                    TRPADR:
023430  104000                                    EMT                             ;SHOULD HAVE TRAPED
                                          ;TRAP TO HERE IF FORCING TRAP BY TESTING 177700
023432  106767  154340                    R0TRAP: MFPS    STATUS
023436  005767  154334                            TST     STATUS                  ;TEST PSW
023442  001401                                    BEQ     1$
023444  104000                                    EMT                             ;NEW PSW SHOULD HAVE BEEN ZERO
023446  026727  155322  023430            1$:     CMP     STBOT-4,#TRPADR ;TEST OLD PC AT STACK
023454  001437                                    BEQ     TRAPB
023456  104000                                    EMT                             ;OLD PC WAS NOT SAVED
                                          ;RETURN HERE ON AN ADDRESS TRAP FROM MEMORY BELOW 28K (OR 30K)
023460  005300                            ATRAP:  DEC     R0                      ;
023462  010067  000032                            MOV     R0,CORH                 ;MOVE THE FIRST NXM LOCATION IN CORH
                                          ;THIS ROUTINE DOES NXM TRAPS UNTIL IT FINDS AN EXISTENT MEMORY LOCATION
023466  013700  023412                            MOV     @#HICORE,R0             ;SET UP THE HIGHEST MEM LOCATION
023472  005300                                    DEC     R0                      ;MAKE 1 LESS THAN THE HIGHEST CORE BOUNDARY
023474  000402                                    BR      NOSUB                   ;DON'T SUBTRACT 1K FIRST TIME
023476  162700  001000                    CTRAP:  SUB     #1000,R0                ;SUBTRACT 1K OCTAL BYTE FROM ADDRESS
                                                                                  ;TO SPEED UP TESTING
023502  012767  023526  154274            NOSUB:  MOV     #BTRAP,4                ;SET UP THE VECTOR
023510  012706  001000                            MOV     #STBOT,SP
023514  005710                                    TST     (R0)                    ;DOES THIS MEMORY EXIST?
                                                                                  ;IF NXM, TRAP TO BTRAP
023516  020027                            DTRAP1: CMP     R0,(PC)+                ;IF EXISTS, IS THIS THE SAME TRAP THAT CAUSED
                                                                                  ;TRAP TO ATRAP
023520  000000                            CORH:   .WORD   0
023522  101414                                    BLOS    TRAPB
023524  104000                                    EMT                             ;CONTENTS OF R0 SHOULD BE LESS THAN OR EQUAL  TO CORH
                                                                                  ;IF THIS COMPARISON FAILS IT MEANS
                                                                                  ;THAT SOME LEGAL ADDRESS TRAPPED, OR
                                                                                  ;THAT AN ILLEGAL ADDRESS DID NOT TRAP
023526  106767  154244                    BTRAP:  MFPS    STATUS
```

```
023532 005767 154240                      TST     STATUS
023536 001401                             BEQ     1$
023540 104000                             EMT                       ;NEW PSW SHOULD HAVE BEEN ZERO
023542 026727 155226 023516       1$:     CMP     STBOT-4,#DTRAP1   ;CHECK IF TRAP PC IS OK
023550 001752                             BEQ     CTRAP
023552                            AUTO1:
023552 104000                             EMT                       ;OLD PC WAS NOT SAVED OR WRONG $TESTN
023554 012767 021336 154222       TRAPB:  MOV     #T04,4            ;RESET TRAP CATCHER
023562 005067 154220                      CLR     6                 ;RESET TRAP CATCHER


                                  ;THIS ROUTINE WILL FIGURE OUT IF YOU HAVE A DL11W
023566 012706 001000                      MOV     #STBOT,SP         ;SET UP THE STACK POINTER
023572 012767 023606 154204               MOV     #NODL,4           ;SET UP THE TRAP VECTOR
023600 005767 153760                      TST     TTCSR             ;TEST THE PUNCH STATUS REGISTER
023604 000405                             BR      DL11W
023606 012767 021336 154170       NODL:   MOV     #T04,4
023614 000167 101036                      JMP     SLU1ST            ;IF NO SLU FIND OUT WHY IN SLU TEST
023620 012767 021336 154156       DL11W:  MOV     #T04,4

                                  ;*****************************************************************************
                                  ;TEST 335         TEST THAT A TTY INTERRUPT CAUSES AN OVERFLOW TRAP
                                  ;*****************************************************************************
023626                            TS335:
023626 012767 000340 154142               MOV     #340,STATUS       ;LOCK OUT INTERRUPT
023634 012706 000400                      MOV     #400,%6           ;SET UP STACK TO OVERFLOW
023640 012767 023702 154136               MOV     #TDEC77,4         ;SET UP OVERFLOW TRAP
023646 016767 154212 001542               MOV     64,     TEMP1     ;SAVE CONTENTS OF INTERRUPT VECTOR
023654 012767 023700 154202               MOV     #TDEC8,64         ;SET UP INTERRUPT VECTOR
023662 012767 000100 153674               MOV     #100,TTCSR        ;SET INTERRUPT ENABLE
023670 005067 154102                      CLR     STATUS            ;ALLOW INTERRUPT TO OCCUR
023674 000167 100756                      JMP     SLU1ST            ;NO INTERRUPT OCCURRED SO GO TO SLU TEST
                                                                    ;TO FIND OUT WHY ADD REPORT PROPER ERROR
023700                            TDEC8:
023700 104000                             EMT                       ;OVERFLOW TRAP DID NOT OCCUR
023702 005067 153656              TDEC77: CLR     TTCSR             ;CLEAR INTERRUPT ENABLE
023706 012767 021336 154070               MOV     #T04,4
023714 005067 154066                      CLR     6
023720 016767 001472 154136               MOV     TEMP1,  64        ;RESTORE CONTENTS OF INTERRUPT VECTOR
                                  ;*****************************************************************************
                                  ;TEST 336         TEST THAT A PENDING INTERRUPT OCCURS BEFORE TRAP
                                  ;*****************************************************************************
023726                            TS336:
023726 012706 001000                      MOV     #STBOT,%6
023732 012767 000340 154036               MOV     #340,STATUS       ;SET TO A HIGH PRIORITY LEVEL
023740 016767 154120 001450               MOV     64,     TEMP1     ;SAVE CONTENTS OF INTERRUPT VECTOR
023746 012767 024012 154110               MOV     #TR0,64
023754 012767 000100 153602               MOV     #100,TTCSR        ;INTERRUPT FOR TTY PUNCH/PRINTER
023762 012767 024014 154044               MOV     #BR71,34          ;TRAP VECTOR
023770 012767 024016 154066               MOV     #TR2,64           ;TTY VECTOR
023776 012767 000340 154032               MOV     #340,36           ;IF TRAP TRAPS, MOVE 340 TO PRIORITY
024004 005067 153766                      CLR     STATUS            ;SHOULD INTERRUPT AT END OF CLR INST
024010 104400                             TRAP                      ;TTY INTERRUPT SHOULD OVERRIDE TRAP
024012                            TR0:
024012 104000                             EMT                       ;TTY SHOULDN'T HAVE INTERRUPTED
024014                            BR71:
```

```
024014  104000                           EMT                       ;TRAP OCCURRED FIRST
024016  005067  154014           TR2:    CLR     36
024022  016767  001370  154034           MOV     TEMP1,  64        ;RESTORE CONTENTS OF INTERRUPT VECTOR
024030  042767  000100  153526           BIC     #100,TTCSR
                                 ;*******************************************************************************
                                 ;TEST 337        TEST THAT A PENDING INTERRUPT, INTERRUPTS BETWEEN TRAPS
                                 ;*******************************************************************************
024036                           TS337:
024036  012706  001000                   MOV     #STBOT,%6
024042  012767  000340  153726           MOV     #340,STATUS
024050  012767  000100  153506           MOV     #100,TTCSR
024056  012767  024124  153750           MOV     #TR3,34           ;TRAP
024064  016767  153774  001324           MOV     64,     TEMP1     ;SAVE CONTENTS OF INTERRUPT VECTOR
024072  012767  024130  153764           MOV     #TR4,64           ;TTY OUTPUT
024100  012767  000340  153760           MOV     #340,66           ;TTY OUTPUT PRIORITY
024106  012767  024126  153704           MOV     #TR5,20           ;IOT
024114  012767  000340  153700           MOV     #340,22           ;IOT PRIORITY
024122  104400                           TRAP                      ;THE ACT OF TRAPPING LOWER PRIORITY
024124  000004                   TR3:    IOT                       ;INTERRUPT SHOULD OCCUR IN PLACE OF IOT TRAP
024126                           TR5:
024126  104000                           EMT                       ;NO INTERRUPT BETWEEN TRAPS,OR WRONG $TSTNM
024130  005067  153666           TR4:    CLR     22                ;CLR IOT PRIORITY
024134  005067  153726                   CLR     66
024140  012767  021350  153666           MOV     #T034,34
024146  016767  001244  153710           MOV     TEMP1,  64        ;RESTORE CONTENTS OF INTERRUPT VECTOR
024154  012767  000022  153636           MOV     #22,20
024162  042767  000100  153374           BIC     #100,TTCSR        ;CLEAR IE BIT IN SLU1 XMIT CSR

                                 ;*******************************************************************************
                                 ;TEST 340        TEST THAT "RESET" GOES TO OUTSIDE WORLD
                                 ;*******************************************************************************
024170                           TS340:
024170  032737  000001  001020           BIT     #1, @#$ENV        ;ARE WE RUNNING UNDER APT
024176  001403                           BEQ     70$               ;IF NO THEN DO TEST
024200  005737  001006                   TST     @#$PASS           ;IS THIS FIRST PASS
024204  001013                           BNE     TS341             ;IF NO THEN SHIP TO NEXT TEST
024206                           70$:
024206  016700  153350                   MOV     TKB,R0            ;MAKE SURE RECEIVER DONE IS CLEAR
024212  012767  000100  153340           MOV     #100,TRCSR        ;SET INTERRUPT ENABLE
024220  000005                           RESET                     ;SHOULD CLEAR INTERRUPT ENABLE
024222  032767  000100  153330           BIT     #100,TRCSR        ;TEST FOR CLEAR
024230  001401                           BEQ     TS341
024232  104000                           EMT                       ;RESET FAILED TO CLEAR TRCSR,OR WRONG $TSTNM
024234                           NODL2:
                                 ;*******************************************************************************
                                 ;TEST 341        TEST THAT RESET HAS NO EFFECT ON THE TRACE TRAP
                                 ;*******************************************************************************
024234                           TS341:
024234  032737  000001  001020           BIT     #1, @#$ENV        ;ARE WE RUNNING UNDER APT
024242  001403                           BEQ     70$               ;IF NO THEN DO TEST
024244  005737  001006                   TST     @#$PASS           ;IS THIS FIRST PASS
024250  001024                           BNE     TS342             ;IF NO THEN SHIP TO NEXT TEST
024252                           70$:
024252  012706  001000                   MOV     #STBOT,%6         ;SET STACK
024256  012767  024304  153530           MOV     #RESET2,14        ;SET UP TRACE VECTOR
024264  012746  000020                   MOV     #20,-(R6)         ;SET THE T-BIT ON STACK
```

```
024270  012746  024276                          MOV     #1$,-(R6)          ;MOVE NEW PC ON STACK
024274  000006                                  RTT
024276  000005                          1$:     RESET                      ;SHOULD HAVE NO EFFECT
024300  000005                                  RESET                      ;NO EFFECT
024302                                  RESET3:
024302  104000                                  EMT                        ;TRACE TRAP FAILED,OR WRONG $TSTNM
024304  005067  153466                  RESET2: CLR     STATUS             ;CLEAR TRACK
024310  005067  153502                          CLR     16                 ;TRACE STATUS
024314  012767  021342  153472                  MOV     #T014,14
024322                                  SKTST2:

                                        ;*************************************************************************
                                        ;TEST 342       TEST THE 'WAIT' INSTRUCTION
                                        ;*************************************************************************
024322                                  TS342:
024322  122767  000001  154470                  CMPB    #APTENV,$ENV               ;RUNING IN APT MODE?
024330  001003                                  BNE     1$                         ;IF NOT, DO THIS TEST
024332  005767  154450                          TST     $PASS                      ;IS THIS THE FIRST PASS?
024336  001051                                  BNE     STP4E                      ;IF NOT FIRST PASS, SKIP TEST
024340                                  1$:
024340  042767  000100  153216                  BIC     #100,TTCSR         ;CLEAR INTERRUPT ENABLE
024346  012706  001000                          MOV     #STBOT,SP          ;SET UP THE STACK
024352  016767  153506  001036                  MOV     64,     TEMP1      ;SAVE CONTENTS OF INTERRUPT VECTOR
024360  012767  024440  153476                  MOV     #WATE,64           ;SET UP THE INTERRUPT VECTOR
024366  005067  153474                          CLR     66
024372  105767  153166                  WATE1:  TSTB    TTCSR              ;WAIT FOR READY
024376  100375                                  BPL     WATE1              ;TO BE UP
024400  012767  000015  153160                  MOV     #15,TPB            ;DO A CARRIAGE RETURN
024406  105767  153152                  WATE2:  TSTB    TTCSR              ;WAIT FOR READY TO COME UP
024412  100375                                  BPL     WATE2
024414  012767  000015  153144                  MOV     #15,TPB            ;DO ANOTHER CARRIAGE RETURN
024422  052767  000100  153134                  BIS     #100,TTCSR         ;SET THE INTERRUPT ENABLE
024430  005067  153342                          CLR     STATUS             ;CLEAR THE PSW
024434  000001                          WATE3:  WAIT                       ;WAIT FOR THE INTERRUPT
024436  104000                                  EMT                        ;WAIT INSTRUCTION DID NOT LOOP
024440  005767  153332                  WATE:   TST     STATUS  ;IS THE PSW CORRECT?
024444  001401                                  BEQ     1$
024446  104000                                  EMT                        ;NEW PSW SHOULD HAVE BEEN ZERO
024450  026727  154320  024436          1$:     CMP     STBOT-4,#WATE3+2           ;IS THE OLD PC SAVED
024456  001401                                  BEQ     STP4E
024460                                  STP4:
024460  104000                                  EMT                        ;OLD PC WAS NOT SAVED OR WRONG $TESTN
024462  016767  000730  153374          STP4E:  MOV     TEMP1,  64         ;RESTORE CONTENTS OF INTERRUPT VECTOR
024470  042767  000100  153066                  BIC     #100,TTCSR         ;CLEAR IE BIT IN SLU1 XMIT CSR
                                        ;*************************************************************************
                                        ;TEST 343       TEST THAT USING REGISTER ADDR (177700) CAUSES TIME OUT.
                                        ;*************************************************************************
024476                                  TS343:

                                                ;
                                                ;REGISTER ADDRESS (177700-177717) CAUSE TIME OUT WHEN USED
                                                ;AS PROGRAM ADDRESS BY THE CPU.
                                                ;
024476  012706  001000                          MOV     #STBOT,SP          ;SET  STACK POINTER
024502  012767  024516  153274                  MOV     #RETR1,RTRAP5      ;SET TRAP RETURN ADDR
024510  005237  177700                  PCN1:   INC     @#177700           ;BAD ADDR REFERENCE, TRAP TO 4
024514  104000                                  EMT                        ;REFERENCING 177700 DID NOT CAUSE TIME OUT
```

```
024516 022767 024514 154250 RETR1:  CMP     #PCN1+4,STBOT-4 ;PROPER PC STORED ON STACK?
024524 001401                       BEQ     TS344
024526 104000                       EMT                     ;OLD PC WAS NOT SAVED IN STACK

                                    ;**************************************************************
                                    ;ODD ADDRESS USED BY A 'WORD'' INSTRUCTION SHOULD NOT
                                    ;CAUSE A TRAP, BUT THE LOW ORDER ADDRESS BIT WOULD BE IGNORED.
                            ;******************************************************************************
                            ;TEST 344       TEST ODD ADDRESS TRAP IS NOT IMPLEMENTED.
                            ;******************************************************************************
024530                      TS344:
                                    ;
024530 012737 024554 000004         MOV     #RETR2,@#RTRAP5 ;SET TRAP RETURN ADDR
024536 005037 000000                CLR     @#0             ;PUT ALL 0 IN LOC 0
024542 005337 000001                DEC     @#1             ;DECREMENT ODD ADDRESS, SHOULD NOT TRAP
024546 022737 177777 000000         CMP     #-1,@#0         ;WORD LOC 0 HAS ALL ONES?
024554                      RETR2:
024554 001401                       BEQ     TS345
024556 104000                       EMT                     ;LOC 0 DID NOT STORE -1,OR ODD ADDR REFERENCE CAUSE TRAP

                                    ;**************************************************************
                                    ;
                                    ;USING ADDRESS 177700 IN MODE 2, CAUSES BUS ERROR, BUT
                                    ;THE REGISTER IN USE WILL BE INCREMENTED.
                                    ;
                            ;******************************************************************************
                            ;TEST 345       TEST THAT IN MODE 2, BAD ADDRESS REFERENCE CAUSES BUS ERROR.
                            ;******************************************************************************
024560                      TS345:
024560 012737 024600 000004         MOV     #RETR3,@#RTRAP5 ;SET TRAP RETURN ADDR
024566 012700 177700                MOV     #177700,R0      ;STORES BAD MEMORY REFERENCE
024572 012720 001234                MOV     #1234,(R0)+     ;BAD ADDR REFERENCE, TRAP TO LOC 4
024576 104000                       EMT                     ;ADDRESSING 177700 DID NOT CAUSE TRAP
024600 022700 177702        RETR3:  CMP     #177702,R0      ;WAS R0 INCREMENTED?
024604 001401                       BEQ     TS346
024606 104000                       EMT                     ;R0 WAS NOT INCREMENTED
                                    ;**************************************************************
                                    ;
                                    ;AFTER THE FIRST BUS ERROR WAS ENCOUNTERED, AN ATTEMPT WAS MADE
                                    ;TO PUSH PC AND PS INTO THE STACK. HOWEVER, IF THE STACK POINTER
                                    ;WAS BAD, A BOUBLE BUS ERROR OCCURED.  THE STACK POINTER WOULD
                                    ;THEN BE SET TO LOCATION 4, OLD PC AND PS WERE PUSHED INTO
                                    ;LOCATIONS 0 AND 2. THE PROCESSOR WOULD TRAP TO 4 AND CONTINUE
                                    ;EXECUTION.
                                    ;
                            ;******************************************************************************
                            ;TEST 346       TEST FOR DOUBLE BUS ERROR.
                            ;******************************************************************************
024610                      TS346:
024610 012767 024652 153166         MOV     #DBE1,RTRAP5    ;SET TRAP RETURN ADDR
024616 012737 000340 000006         MOV     #340,@#6        ;SET UP PS
024624 012767 024650 153156         MOV     #DBE2,RTRAP     ;SET TRAP RETURN ADDR
024632 012737 000340 000012         MOV     #340,@#12       ;SET UP PS
024640 012706 177700                MOV     #177700,SP      ;SET ILLEGAL SP
024644 000077               DBE:    TRAPA                   ;ILLEGAL INSTRUCTION
024646 104000                       EMT                     ;DOUBLE BUS ERROR DID NOT CAUSE TRAP
```

```
024650                             DBE2:
024650  104000                             EMT                       ;TRAP TO WRONG LOCATION
024652  022737  024646  000000     DBE1:   CMP     #DBE+2,@#0        ;OLD PC GOT SAVED?
024660  001401                             BEQ     DBE3
024662  104000                             EMT                       ;OLD PC DID NOT GET SAVEDD
024664  022737  000340  000002     DBE3:   CMP     #340,@#2          ;CORRECT PS SAVED?
024672  001401                             BEQ     DBE4
024674  104000                             EMT                       ;CORRECT PS DID NOT GET SAVE
024676  022706  000000             DBE4:   CMP     #0,SP             ;SP POINTS TO LOC 0?
024702  001401                             BEQ     DBE5
024704  104000                             EMT                       ;SP IS NOT POINTING TO LOC 0
024706  012706  001000             DBE5:   MOV     #STBOT,SP         ;RESET SP
024712  012767  021336  153064            MOV     #T04,4            ;RESET VECTOR 4
024720  012767  021340  153062            MOV     #T010,10          ;RESET VECTOR 10

                                           ;*************************************************************
                                           ;THIS TEST WILL CHECK THE SERVICE ROUTINE FOR A CONTROL CHIP ERROR.
                                           ;THIS IS DONE BY EXECUTING INSTRUCTIONS WHICH JUMP TO NON-EXISTENT
                                           ;CONTROL-CHIP. THE TEST EXECUTES AN FIS INSTRUCTION WHICH
                                           ;IS ILLEGAL ON ALL PROCESSORS USING THE DCF11-A CHIP SET.
                                           ;A CTLERR TRAPS TO LOCATION 10.
                                           ;THE RESET LINE IS ALSO ASSERTED FOR 1 CYCLE.
                                   ;***********************************************************************
                                   ;TEST 347          TEST CTLERR SERVICE ROUTINE
                                   ;***********************************************************************
024726                             TS347:

024726  012706  001000                     MOV     #STBOT,R6         ;INIT STACK POINTER
024732  012737  024752  000010             MOV     #1$,@#10                  ;SET UP RETURN ADDR FROM TRAP
024740  012737  000340  000012             MOV     #340,@#12         ;SET TRAP PRIORITY=7
024746  075006                             FADD    R6                ;EXECUTE FIS INSTR..SHOULD CAUSE CTLERR
024750  000000                             HALT                      ;DID NOT TRAP..CHECK CSEL LINE
024752  012706  001000             1$:     MOV     #STBOT,R6         ;RE-INIT STACK POINTER
024756  012767  021340  153024             MOV     #T010,10          ;RESET VECTOR 10

                                   ;***********************************************************************
                                   ;TEST 350          TEST THAT ALL RESERVED INSTRUCTIONS TRAP
                                   ;***********************************************************************
024764                             TS350:
024764  042767  000100  152572             BIC     #100,TTCSR
                                                                     ; SET UP TO SEE IF
024772  013767  000010  000042             MOV     @#10,TENSAVE      ;  THIS PROCESSOR HAS THE
025000  012737  025044  000010             MOV     #TRAP10,@#10      ;  FLOATING POINT OPTION
025006  170127  000000                     LDFPS   #0                ;DO A FPP INSTRUCTION
                                                                     ;IF NO TRAP FPP INSTALLED
025012  013767  025366  000356             MOV     @#FPP,FINISH      ;SO RESET END OF TABLE POINTER
025020  000411                             BR      AROUND            ; THE FOLLOWING

                                   ;* IF NO CIS OPTION TRAP TO HERE
025022  042777  000040  174354     CISTRP: BIC     #40,@SWR          ;CLEAR CIS OPTION IN SWR
025030  012716  025076                     MOV     #CONCIS,(SP)      ;CHANGE RETURN ADDRESS TO CONCIS LOCATION
025034  000002                             RTI                       ;RETURN
025036  000000                     CISADR: .WORD   0                 ;DATA FOR CIS
025040  000000                             .WORD   0                 ;                 INSTRUCTION
025042  000000                     TENSAVE:.WORD   0                 ; A PLACE TO STORE CONTENTS OF 10
025044                             TRAP10:                           ; LEAVE THE TABLE ALONE
```

```

025044                                AROUND:                                   ;  CONTINUATION POINT
025044  012737  000246  000244                  MOV     #246,@#244              ;  RESTORE THE TRAP VECTOR
025052  012737  025022  000010                  MOV     #CISTRP,@#10            ;SET UP TO SEE IF THIS HAS THE CIS OPTION
025060  076144                                  .WORD   076144                  ;EXECUTE A CMPCI INSTRUCTION
025062  025036                                  .WORD   CISADR                  ;OPERANDS
025064  025036                                  .WORD   CISADR                  ;        FOR CIS
025066  000000                                  .WORD   0                       ;              INSTRUCTION
025070  052777  000040  174306                  BIS     #40,@SWR                ;SET CIS PRESENT BIT
025076  016737  177740  000010        CONCIS:   MOV     TENSAVE,@#10            ; RESTORE THE ILLEGAL INST. VECTOR
025104  012703  025256                          MOV     #TABLE,TAB              ;TABLE POINTER
025110  012305                        GIN1:     MOV     (TAB)+,FIRST            ;FIRST OR CURRENT INSTRUCTION
025112  012301                                  MOV     (TAB)+,LAST             ;LAST INSTRUCTION OR GROUP
025114  020537  025332                          CMP     FIRST,@#CIS
025120  001007                                  BNE     1$
025122  032777  000040  174254                  BIT     #40,@SWR
025130  001403                                  BEQ     1$
025132  012703  025366                          MOV     #FPP,TAB
025136  000764                                  BR      GIN1
025140  020567  000232                1$:       CMP     FIRST,FINISH            ;TESTED ALL
025144  001415                                  BEQ     GIN3                    ;YES BRANCH
025146  010567  000226                          MOV     FIRST,INST              ;SET UP INST
025152  005267  000222                GIN2:     INC     INST
025156  012767  025212  152624                  MOV     #RET,10                 ;SET UP RETURN FROM TRAP
025164  012706  001000                          MOV     #STBOT,SP               ;SET UP STACK POINTER
025170  005067  152602                          CLR     CC                      ;CLEAR PRIORITY
025174  000167  000200                          JMP     INST                    ;EXECUTE RESERVED INSTRUCTION
025200  012767  021340  152602        GIN3:     MOV     #T010,10                ;RESET VECTOR 10
025206  000167  000252                          JMP     THRPRT                  ;JUMP TO EIS TEST


                                      ;TRAPPING SHOULD SEND YOU HERE
025212  020627  000774                RET:      CMP     SP,#STBOT-4             ;TEST DECREMENT OF SP
025216  001401                                  BEQ     RET1
025220  104000                                  EMT                             ;WRONG DECREMENT
025222  026727  153546  025402        RET1:     CMP     STBOT-4,#INST+2 ;LOC OF INST UNINCREMENTED
025230  001401                                  BEQ     RET2
025232  104000                                  EMT                             ;INST INC ON TRAP
025234  005767  153536                RET2:     TST     STBOT-2
025240  001401                                  BEQ     RET3
025242                                RET4:
025242  104000                                  EMT                             ;CONDITION CODES SET ON TRAP OR WRONG $TSTNM
025244  026701  000130                RET3:     CMP     INST,LAST
025250  001717                                  BEQ     GIN1                    ;SET UP NEW GROUP
025252  000167  177674                          JMP     GIN2                    ;FINISH OLD GROUP
                                                                                ;END OF INSTRUCTION GROUP
025256  000007                        TABLE:    7                               ;END OF OPERATE
025260  000077                                  77
025262  000207                                  207                             ;RTS,RT1,JMP
025264  000227                                  227
025266  006777                                  6777
025270  007777                                  7777
025272  075037                                  075037
025274  076017                                  76017
025276  076032                                  76032
025300  076037                                  76037
025302  076045                                  76045
```

```
025304 076047                   76047
025306 076077                   76077
025310 076127                   76127
025312 076132                   76132
025314 076137                   76137
025316 076145                   76145
025320 076147                   76147
025322 076157                   76157
025324 076167                   76167
025326 076177                   76177
025330 076777                   76777
025332 076017           CIS:    76017
025334 076032                   76032
025336 076037                   76037
025340 076045                   76045
025342 076047                   76047
025344 076077                   76077
025346 076127                   76127
025350 076132                   76132
025352 076137                   76137
025354 076145                   76145
025356 076147                   76147
025360 076157                   76157
025362 076167                   76167
025364 076177                   76177
025366 167777           FPP:    167777                  ;  START OF THE FPP INSTRUCTIONS
025370 177700                   177700
025372 177716                   177716
025374 177777                   177777
025376 025376           FINISH: .                       ;END FLAG
025400 000000           INST:   HALT                    ;WILL CONTINUE RESERVED INST
025402 000000                   HALT                    ;SHOULD TRAP TO LOC 10
025404 000000                   HALT                    ;LOC 10 SHOULD SEND YOU TO
025406 000000                   HALT                    ;RET
025410 000000                   HALT
                        .SBTTL ** STARTING OF EIS TEST **



       000000           DUMMY=  0
       000051           F=      51
       000176           N=      176

025412                  COUNT:
       025414                   .=COUNT+2
025414                  PSWORD:
       025416                   .=PSWORD+2
025416                  TEMP1:
       025420                   .=TEMP1+2
025420                  TEMP2:
       025422                   .=TEMP2+2
025422                  TEMP3:
       025424                   .=TEMP3+2
025424                  TEMP4:
       025426                   .=TEMP4+2
025426 000000           TEMP5:  .WORD
025430 000000           TEMP6:  .WORD
```

```
025432  177771          S1:     -7
025434  025432          S2:     S1
025436  177772          S3:     -6
025440  177777          S4:     -1
025442  040000          S5:     40000
025444  025442          S6:     S5
025446  040000          S7:     40000
025450  177776          S8:     -2
025452  000002          S9:     2
025454  025452          S10:    S9
025456  000002          S11:    2
025460  177566          $TPB:   177566
025462  177564          $TPS:   177564
```

```
025464                                    THRPRT:
025464  012705  001004                            MOV     #$TESTN,R5          ;MAKE R5 POINT TO WHERE TEST # IS SAVED
025470  005037  025412                            CLR     @#COUNT             ;CLEAR THE COUNTER
025474  012715  000001                            MOV     #1,(R5)             ;INITIALIZE TEST NUMBER
025500  012706  001000                            MOV     #STBOT,SP           ;** STACK AT STBOT **
025504  012737  000001  025416    2$:             MOV     #1,@#TEMP1          ;TEMP1=1
025512  005037  025420                            CLR     @#TEMP2             ;TEMP2=0
025516  012737  000001  025422                    MOV     #1,@#TEMP3          ;TEMP3=1
025524  005037  025424                            CLR     @#TEMP4             ;TEMP4=0
025530  106427  000000                            MTPS    #0
```

```
                                                     ;**********************************
                                                     ;      ASH INSTRUCTION TESTS
                                                     ;**********************************

                                                             ;*************
                                                             ; TESTS 1-36
                                                             ;*************

025534  013700  025416          ASTART: MOV     @#TEMP1,%0              ;LOAD R0 WITH THE CONTENTS OF TEMP1
025540  032737  000001  001006          BIT     #1,@#$PASS              ;IS IT AN EVEN PASS ?
025546  001004                          BNE     2$                      ;IF NOT THEN GO TO 2$
025550  013701  025420                  MOV     @#TEMP2,R1              ;OTHERWISE EXECUTE THE INSTRUCTION
                                                                        ;IN MODE 0 USING R1
025554  072001                          ASH     R1,R0
025556  000402                          BR      4$
025560  072067  177634          2$:     ASH     TEMP2,%0                ;SHIFT R0 BY THE NUMBER SPECIFIED BY TEMP2
025564  106737  025414          4$:     MFPS    @#PSWORD                ;SAVE PS
025570  123737  025424  025414          CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
025576  001401                          BEQ     11$
025600  104000                          EMT                             ;THE PS IS NOT EQUAL TO 0
025602  005237  025412          11$:    INC     @#COUNT                 ;INCREMENT THE COUNTER
025606  023700  025422                  CMP     @#TEMP3,%0              ;IS THE RESULT IN R0 EQUAL TO TEMP3?
025612  001401                          BEQ     12$
025614                          6$:
025614  104000                          EMT                             ;EITHER INCORRECT R0 OR INCORRECT SEQUENCE
025616  021537  025412          12$:    CMP     (R5),@#COUNT            ;IS THE TEST NUMBER EQUAL TO THE
                                                                        ;COUNTER?
025622  001374                          BNE     6$                      ;IF NOT GO TO THE HLT ABOVE
025624  005215                          INC     (R5)
025626  021527  000037                  CMP     (R5),#37                ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT
                                                                        ;BY 14. AND RIGHT BY 14.?
025632  002011                          BGE     8$
025634  005237  025420                  INC     @#TEMP2
025640  006367  177556                  ASL     TEMP3                   ;SHIFT TEMP3 LEFT.
025644  021527  000020                  CMP     (R5),#20                ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
025650  001004                          BNE     REGR1
025652  000167  000670                  JMP     NEGAT                   ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
025656  004767  000712          8$:     JSR     PC,TST37                ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
025662  013701  025416          REGR1:  MOV     @#TEMP1,%1              ;LOAD R1 WITH THE CONTENTS OF TEMP1
025666  032737  000001  001006          BIT     #1,@#$PASS              ;IS IT AN EVEN PASS ?
025674  001004                          BNE     2$                      ;IF NOT THEN GO TO 2$
025676  013702  025420                  MOV     @#TEMP2,R2              ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
025702  072102                          ASH     R2,R1                   ;USING R1
025704  000402                          BR      4$
025706  072167  177506          2$:     ASH     TEMP2,%1                ;SHIFT R1 BY THE NUMBER SPECIFIED BY TEMP2
025712  106737  025414          4$:     MFPS    @#PSWORD                ;SAVE PS
025716  123737  025424  025414          CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
025724  001401                          BEQ     11$
025726  104000                          EMT                             ;THE PS IS NOT EQUAL TO 0
025730  005237  025412          11$:    INC     @#COUNT                 ;INCREMENT THE COUNTER
025734  023701  025422                  CMP     @#TEMP3,%1              ;IS THE RESULT IN R1 EQUAL TO TEMP3?
```

```
025740 001401                              BEQ     12$
025742                             6$:
025742 104000                              EMT                     ;EITHER INCORRECT R1 OR INCORRECT SEQUENCE
025744 021537 025412               12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
025750 001374                              BNE     6$              ;IF NOT GO TO THE HLT ABOVE
025752 005215                              INC     (R5)
025754 021527 000037                       CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT
                                                                   ;BY 14. AND RIGHT BY 14.?
025760 002011                              BGE     8$
025762 005237 025420                       INC     @#TEMP2
025766 006367 177430                       ASL     TEMP3           ;SHIFT TEMP3 LEFT
025772 021527 000020                       CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
025776 001004                              BNE     REGR2
026000 000167 000542                       JMP     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
026004 004767 000564               8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
026010 013702 025416               REGR2:  MOV     @#TEMP1,%2      ;LOAD R2 WITH THE CONTENTS OF TEMP1
026014 032737 000001 001006                BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
026022 001004                              BNE     2$              ;IF NOT THEN GO TO 2$
026024 013703 025420                       MOV     @#TEMP2,R3      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
026030 072203                              ASH     R3,R2           ;USING R2
026032 000402                              BR      4$
026034 072267 177360               2$:     ASH     TEMP2,%2        ;SHIFT R2 BY THE NUMBER SPECIFIED BY TEMP2
026040 106737 025414               4$:     MFPS    @#PSWORD        ;SAVE PS
026044 123737 025424 025414                CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
026052 001401                              BEQ     11$
026054 104000                              EMT                     ;THE PS IS NOT EQUAL TO 0
026056 005237 025412               11$:    INC     @#COUNT
026062 023702 025422                       CMP     @#TEMP3,%2      ;IS THE RESULT IN R2 EQUAL TO TEMP3?
026066 001401                              BEQ     12$
026070                             6$:
026070 104000                              EMT                     ;EITHER INCORRECT R2 OR INCORRECT SEQUENCE
026072 021537 025412               12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
026076 001374                              BNE     6$              ;IF NOT GO TO THE HLT ABOVE
026100 005215                              INC     (R5)
026102 021527 000037                       CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
                                                                   ;LEFT BY 14, AND RIGHT BY 14.?
026106 002011                              BGE     8$
026110 005237 025420                       INC     @#TEMP2
026114 006367 177302                       ASL     TEMP3           ;SHIFTED TEMP3 LEFT
026120 021527 000020                       CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
026124 001004                              BNE     REGR3
026126 000167 000414                       JMP     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
026132 004767 000436               8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
026136 013703 025416               REGR3:  MOV     @#TEMP1,%3      ;LOAD R3 WITH THE CONTENTS OF TEMP1
026142 032737 000001 001006                BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
026150 001004                              BNE     2$              ;IF NOT THEN GO TO 2$
026152 013704 025420                       MOV     @#TEMP2,R4      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
026156 072304                              ASH     R4,R3           ;USING R3
026160 000402                              BR      4$
026162 072367 177232               2$:     ASH     TEMP2,%3        ;SHIFT R3 BY THE NUMBER SPECIFIED BY TEMP2
026166 106737 025414               4$:     MFPS    @#PSWORD        ;SAVE PS
026172 123737 025424 025414                CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
026200 001401                              BEQ     11$
026202 104000                              EMT                     ;THE PS IS NOT EQUAL TO 0.
026204 005237 025412               11$:    INC     @#COUNT
026210 023703 025422                       CMP     @#TEMP3,%3      ;IS THE RESULT IN R3 EQUAL TO TEMP3?
```

```
026214  001401                          BEQ     12$
026216                          6$:
026216  104000                          EMT                     ;EITHER INCORRECT R3 OR INCORRECT SEQUENCE
026220  021537  025412          12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
026224  001374                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
026226  005215                          INC     (R5)
026230  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
                                                                ;LEFT BY 14, AND RIGHT BY 14.?
026234  002010                          BGE     8$
026236  005237  025420                  INC     @#TEMP2
026242  006367  177154                  ASL     TEMP3           ;SHIFT TEMP3 LEFT?
026246  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
026252  001003                          BNE     REGR4
026254  000534                          BR      NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
026256  004767  000312          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
026262  013704  025416          REGR4:  MOV     @#TEMP1,%4      ;LOAD R4 WITH THE CONTENTS OF TEMP1
026266  010501                          MOV     R5,R1           ;SAVE R5
026270  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
026276  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
026300  013705  025420                  MOV     @#TEMP2,R5      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
026304  072405                          ASH     R5,R4           ;USING R4
026306  000402                          BR      4$
026310  072467  177104          2$:     ASH     TEMP2,%4        ;SHIFT R4 BY THE NUMBER SPECIFIED BY TEMP2
026314  106737  025414          4$:     MFPS    @#PSWORD        ;SAVE PS
026320  123737  025424  025414          CMPB    @#TEMP4,@#PSWORD;IS PS = TEMP4 ?
026326  001401                          BEQ     11$
026330  104000                          EMT                     ;THE PS IS NOT EQUAL TO 0
026332  005237  025412          11$:    INC     @#COUNT
026336  023704  025422                  CMP     @#TEMP3,%4      ;IS THE RESULT IN R4 EQUAL TO TEMP3?
026342  001401                          BEQ     12$
026344                          6$:
026344  104000                          EMT                     ;EITHER INCORRECT R4 OR INCORRECT SEQUENCE
026346  010105                  12$:    MOV     R1,R5           ;RESTORE R5
026350  021537  025412                  CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
026354  001373                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
026356  005215                          INC     (R5)
026360  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN
                                                                ;SHIFTED LEFT BY 14. AND RIGHT BY 14.?
026364  002010                          BGE     8$
026366  005237  025420                  INC     @#TEMP2
026372  006367  177024                  ASL     TEMP3           ;SHIFT TEMP3 LEFT
026376  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTER BEEN SHIFTED BY 14.?
026402  001003                          BNE     REGR5
026404  000460                          BR      NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
026406  004767  000162          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
026412  010501                  REGR5:  MOV     R5,R1           ;SAVE R5
026414  013705  025416                  MOV     @#TEMP1,%5      ;LOAD R5 WITH THE CONTENTS OF TEMP1
026420  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
026426  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
026430  013700  025420                  MOV     @#TEMP2,R0      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
026434  072500                          ASH     R0,R5           ;USING R5
026436  000402                          BR      4$
026440  072567  176754          2$:     ASH     TEMP2,%5        ;SHIFT R5 BY THE NUMBER SPECIFIED BY TEMP2
026444  106737  025414          4$:     MFPS    @#PSWORD        ;SAVE PS
026450  123737  025424  025414          CMPB    @#TEMP4,@#PSWORD;IS PS = TEMP4 ?
026456  001401                          BEQ     11$
```

```
026460  104000                          EMT                             ;THE PS IS NOT EQUAL TO 0.
026462  005237  025412          11$:    INC     @#COUNT
026466  023705  025422                  CMP     @#TEMP3,%5              ;IS THE RESULT IN R5 EQUAL TO TEMP3?
026472  001401                          BEQ     12$
026474                          6$:
026474  104000                          EMT                             ;EITHER INCORRECT R5 OR INCORRECT SEQUENCE
026476  021137  025412          12$:    CMP     (R1),@#COUNT            ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
026502  001374                          BNE     6$                      ;IF NOT GO TO THE HLT ABOVE
026504  010105                          MOV     R1,R5                   ;RESTORE R5
026506  005215                          INC     (R5)
026510  021527  000037                  CMP     (R5),#37                ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
                                                                        ;LEFT BY 14. AND RIGHT BY 14.?
026514  002010                          BGE     8$                      ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
026516  005237  025420                  INC     @#TEMP2
026522  006367  176674                  ASL     TEMP3                   ;SHIFT TEMP3 LEFT
026526  021527  000020                  CMP     (R5),#20                ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
026532  001405                          BEQ     NEGAT                   ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
026534  000402                          BR      10$
026536  004767  000032          8$:     JSR     PC,TST37
026542  000167  176766          10$:    JMP     ASTART                  ;GO BACK TO START
026546  012737  040000  025416  NEGAT:  MOV     #40000,@#TEMP1          ;TEMP1=40000
026554  012737  177762  025420          MOV     #177762,@#TEMP2         ;TEMP2=177762
026562  012737  000001  025422          MOV     #1,@#TEMP3              ;TEMP3=1
026570  000167  176740                  JMP     ASTART
026574  021527  000037          TST37:  CMP     (R5),#37                ;IS IT TEST 37?
026600  001013                          BNE     TST40                   ;IF NOT THEN TRY TEST 40
026602  005037  025416                  CLR     @#TEMP1                 ;0
026606  012737  000020  025420          MOV     #16.,@#TEMP2            ;SHIFTED BY 16
026614  005037  025422                  CLR     @#TEMP3                 ;IS=0
026620  012737  000004  025424          MOV     #4,@#TEMP4              ;AND PS=4
026626  000207                          RTS     PC
026630  021527  000040          TST40:  CMP     (R5),#40                ;IS IT TEST 40?
026634  001003                          BNE     TST41                   ;IF NOT THEN TRY TEST 41
026636  005037  025420                  CLR     @#TEMP2                 ;0 SHIFTED BY 0=0 AND PS=4
026642  000207                          RTS     PC
026644  021527  000041          TST41:  CMP     (R5),#41                ;IS IT TEST 41?
026650  001004                          BNE     TST42                   ;IF NOT THEN TRY TEST 42
026652  012737  177760  025420          MOV     #-16.,@#TEMP2           ;0 SHIFTED BY -16.=0 AND PS=4
026660  000207                          RTS     PC
026662  021527  000042          TST42:  CMP     (R5),#42                ;IS IT TEST 42?
026666  001013                          BNE     TST43                   ;IF NOT THEN TRY TEST 43
026670  012737  100000  025416          MOV     #100000,@#TEMP1         ;100000
026676  005237  025420                  INC     @#TEMP2                 ;SHIFTED BY -15
026702  005337  025422                  DEC     @#TEMP3                 ;IS=-1
026706  012737  000010  025424          MOV     #10,@#TEMP4                     ;AND PS=10
026714  000207                          RTS     PC
026716  021527  000043          TST43:  CMP     (R5),#43                ;IS IT TEST 43?
026722  001012                          BNE     TST44                   ;IF NOT THEN IF NOT THEN TRY TEST 44
026724  012737  125252  025416          MOV     #125252,@#TEMP1         ;125252
026732  012737  177777  025420          MOV     #-1,@#TEMP2             ;SHIFTED BY -1
026740  012737  152525  025422          MOV     #152525,@#TEMP3         ;IS=152525 AND PS=10
026746  000207                          RTS     PC
026750  021527  000044          TST44:  CMP     (R5),#44                ;IS IT TEST 44?
026754  001012                          BNE     TST45                   ;IF NOT THEN TRY TEST 45
026756  012737  000001  025420          MOV     #1,@#TEMP2              ;125252 SHIFTED BY 1
026764  012737  052524  025422          MOV     #52524,@#TEMP3          ;IS=52524
```

```
026772  012737  000003  025424            MOV     #3,@#TEMP4        ;AND PS=3
027000  000207                            RTS     PC
027002  021527  000045          TST45:    CMP     (R5),#45          ;IS IT TEST 45?
027006  001012                            BNE     TST46             ;IF NOT THEN TRY TEST 46
027010  012737  177776  025420            MOV     #-2,@#TEMP2       ;125252 SHIFTED BY -2
027016  012737  165252  025422            MOV     #165252,@#TEMP3   ;IS=165252
027024  012737  000011  025424            MOV     #11,@#TEMP4       ;AND PS=11
027032  000207                            RTS     PC
027034  021527  000046          TST46:    CMP     (R5),#46          ;IS IT TEST 46?
027040  001014                            BNE     TST47             ;IF NOT THEN TRY TEST 47
027042  012737  177777  025416            MOV     #-1,@#TEMP1       ;-1
027050  012737  000020  025420            MOV     #16.,@#TEMP2      ;SHIFTED BY 15.
027056  005037  025422                    CLR     @#TEMP3           ;IS=0
027062  012737  000007  025424            MOV     #7,@#TEMP4        ;AND PS=7
027070  000207                            RTS     PC
027072  021527  000047          TST47:    CMP     (R5),#47          ;IS IT TEST 47?
027076  001011                            BNE     TST50             ;IF NOT THEN TRY TEST 50
027100  005337  025420                    DEC     @#TEMP2           ;-1 SHIFTED BY 15
027104  012737  100000  025422            MOV     #100000,@#TEMP3   ;IS=100000
027112  012737  000011  025424            MOV     #11,@#TEMP4       ;AND PS=11
027120  000207                            RTS     PC
027122  021527  000050          TST50:    CMP     (R5),#50          ;IS IT TEST 50
027126  001007                            BNE     ENT51             ;IF NOT THEN TRY TEST 51
027130  012737  137777  025416            MOV     #137777,@#TEMP1   ;137777 SHIFTED BY 15. IS=100000
027136  012737  000013  025424            MOV     #13,@#TEMP4       ;AND PS=13
027144  000207                            RTS     PC
027146  021527  000051          ENT51:    CMP     (R5),#51          ;IS IT ENTERING TEST 51?
027152  001401                            BEQ     1$
027154  104000                            EMT                       ;TEST NUMBER GOOFED

027156  005726                  1$:       TST     (SP)+             ;RESTORE STACK POINTER
027160  012704  177771                    MOV     #-7,%4
027164  012702  025432                    MOV     #S1,%2
027170  012703  025434                    MOV     #S2,%3
                                ;*****************************************************************
                                ;TEST:51     11/34    ASH     125252 SHIFTED BY #5 = 52500   PS = 3
                                ;*****************************************************************

027174  012701  125252          TST51:    MOV     #125252,%1        ;LOAD R1 WITH 125252
027200  072127  000005                    ASH     #5,%1             ;SHIFT R1 BY #5
027204  106737  025414                    MFPS    @#PSWORD          ;SAVE PS
027210  122737  000003  025414            CMPB    #3,@#PSWORD       ;IS THE PS 3?
027216  001401                            BEQ     11$
027220  104000                            EMT                       ;THE PS IS NOT EQUAL TO 3
027222  022701  052500          11$:      CMP     #52500,%1         ;IS THE RESULT 52500?
027226  001401                            BEQ     12$
027230  104000                            EMT                       ;R1 IS NOT EQUAL TO 52500 OR INCORRECT SEQUENCE
027232  005215                  12$:      INC     (R5)



                                ;*****************************************************************
                                ;TEST:52     11/34    ASH     125252 SHIFTED BY @S2 = 177525   PS = 10
                                ;*****************************************************************

027234  012700  125252          TST52:    MOV     #125252,%0        ;LOAD R0 WITH 125252
```

```
027240  072077  176170                   ASH     @S2,%0              ;SHIFT R0 BY @S2
027244  106737  025414                   MFPS    @#PSWORD            ;SAVE PS
027250  122737  000010  025414           CMPB    #10,@#PSWORD        ;IS THE PS 10?
027256  001401                           BEQ     11$
027260  104000                           EMT                         ;THE PS IS NOT EQUAL TO 10
027262  022700  177525          11$:     CMP     #177525,%0          ;IS THE RESULT 177525?
027266  001401                           BEQ     12$
027270  104000                           EMT                         ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027272  005215                  12$:     INC     (R5)



                                ;*******************************************************************
                                ;TEST:53    11/34    ASH      125252 SHIFTED BY @#S1 = 177525   PS = 10
                                ;*******************************************************************

027274  012700  125252          TST53:   MOV     #125252,%0          ;LOAD R0 WITH 125252
027300  072037  025432                   ASH     @#S1,%0             ;SHIFT R0 BY @#S1
027304  106737  025414                   MFPS    @#PSWORD            ;SAVE PS
027310  122737  000010  025414           CMPB    #10,@#PSWORD        ;IS THE PS 10?
027316  001401                           BEQ     11$
027320  104000                           EMT                         ;THE PS IS NOT EQUAL TO 10
027322  022700  177525          11$:     CMP     #177525,%0          ;IS THE RESULT 177525?
027326  001401                           BEQ     12$
027330  104000                           EMT                         ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027332  005215                  12$:     INC     (R5)



                                ;*******************************************************************
                                ;TEST:54    11/34    ASH      125252 SHIFTED BY (2) = 177525   PS = 10
                                ;*******************************************************************

027334  012700  125252          TST54:   MOV     #125252,%0          ;LOAD R0 WITH 125252
027340  072012                           ASH     (2),%0              ;SHIFT R0 BY (2)
027342  106737  025414                   MFPS    @#PSWORD            ;SAVE PS
027346  122737  000010  025414           CMPB    #10,@#PSWORD        ;IS THE PS 10?
027354  001401                           BEQ     11$
027356  104000                           EMT                         ;THE PS IS NOT EQUAL TO 10
027360  022700  177525          11$:     CMP     #177525,%0          ;IS THE RESULT 177525?
027364  001401                           BEQ     12$
027366  104000                           EMT                         ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027370  005215                  12$:     INC     (R5)



                                ;*******************************************************************
                                ;TEST:55    11/34    ASH      125252 SHIFTED BY (2)+ = 177525   PS = 10
                                ;*******************************************************************

027372  012700  125252          TST55:   MOV     #125252,%0          ;LOAD R0 WITH 125252
027376  072022                           ASH     (2)+,%0             ;SHIFT R0 BY (2)+
027400  106737  025414                   MFPS    @#PSWORD            ;SAVE PS
027404  122737  000010  025414           CMPB    #10,@#PSWORD        ;IS THE PS 10?
027412  001401                           BEQ     11$
027414  104000                           EMT                         ;THE PS IS NOT EQUAL TO 10
```

```
027416  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
027422  001401                          BEQ     12$
027424  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027426  005215                  12$:    INC     (R5)



                                ;*************************************************************************
                                ;TEST:56    11/34    ASH      125252 SHIFTED BY -(2) = 177525  PS = 10
                                ;*************************************************************************

027430  012700  125252          TST56:  MOV     #125252,%0      ;LOAD R0 WITH 125252
027434  072042                          ASH     -(2),%0         ;SHIFT R0 BY -(2)
027436  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
027442  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
027450  001401                          BEQ     11$
027452  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
027454  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
027460  001401                          BEQ     12$
027462  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027464  005215                  12$:    INC     (R5)



                                ;*************************************************************************
                                ;TEST:57    11/34    ASH      125252 SHIFTED BY 2(3) = 177252  PS = 11
                                ;*************************************************************************

027466  012700  125252          TST57:  MOV     #125252,%0      ;LOAD R0 WITH 125252
027472  072063  000002                  ASH     2(3),%0         ;SHIFT R0 BY 2(3)
027476  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
027502  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS THE PS 11?
027510  001401                          BEQ     11$
027512  104000                          EMT                     ;THE PS IS NOT EQUAL TO 11
027514  022700  177252          11$:    CMP     #177252,%0      ;IS THE RESULT 177252?
027520  001401                          BEQ     12$
027522  104000                          EMT                     ;R0 IS NOT EQUAL TO 177252 OR INCORRECT SEQUENCE
027524  005215                  12$:    INC     (R5)



                                ;*************************************************************************
                                ;TEST:60    11/34    ASH      125252 SHIFTED BY @(3) = 177525  PS = 10
                                ;*************************************************************************

027526  012700  125252          TST60:  MOV     #125252,%0      ;LOAD R0 WITH 125252
027532  072073  000000                  ASH     @(3),%0         ;SHIFT R0 BY @(3)
027536  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
027542  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
027550  001401                          BEQ     11$
027552  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
027554  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
027560  001401                          BEQ     12$
027562  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
027564  005215                  12$:    INC     (R5)
```

```



                                        ;*********************************************************************
                                        ;TEST:61    11/34   ASH     125252 SHIFTED BY @(3)+ = 177525  PS = 10
                                        ;*********************************************************************

 027566  012700  125252                 TST61:  MOV     #125252,%0      ;LOAD R0 WITH 125252
 027572  072033                                 ASH     @(3)+,%0        ;SHIFT R0 BY @(3)+
 027574  106737  025414                         MFPS    @#PSWORD        ;SAVE PS
 027600  122737  000010  025414                 CMPB    #10,@#PSWORD    ;IS THE PS 10?
 027606  001401                                 BEQ     11$
 027610  104000                                 EMT                     ;THE PS IS NOT EQUAL TO 10
 027612  022700  177525                 11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
 027616  001401                                 BEQ     12$
 027620  104000                                 EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
 027622  005215                         12$:    INC     (R5)




                                        ;*********************************************************************
                                        ;TEST:62    11/34   ASH     125252 SHIFTED BY @-(3) = 177525  PS = 10
                                        ;*********************************************************************

 027624  012700  125252                 TST62:  MOV     #125252,%0      ;LOAD R0 WITH 125252
 027630  072053                                 ASH     @-(3),%0        ;SHIFT R0 BY @-(3)
 027632  106737  025414                         MFPS    @#PSWORD        ;SAVE PS
 027636  122737  000010  025414                 CMPB    #10,@#PSWORD    ;IS THE PS 10?
 027644  001401                                 BEQ     11$
 027646  104000                                 EMT                     ;THE PS IS NOT EQUAL TO 10
 027650  022700  177525                 11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
 027654  001401                                 BEQ     12$
 027656  104000                                 EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
 027660  005215                         12$:    INC     (R5)



```

```
                                                          ;************************************
                                                          ;       ASHC INSTRUCTION TESTS
                                                          ;************************************



                                                                  ;*************
                                                                  ;TESTS 63-157
                                                                  ;*************


027662  012737  000062  025412            MOV     #62,@#COUNT
027670  005037  025416                    CLR     @#TEMP1             ;TEMP1=0
027674  012737  000001  025420            MOV     #1,@#TEMP2          ;TEMP2=1
027702  005037  025422                    CLR     @#TEMP3             ;TEMP3=0
027706  005037  025424                    CLR     @#TEMP4             ;TEMP4=0
027712  012737  000001  025426            MOV     #1,@#TEMP5          ;TEMP5=1
027720  005037  025430                    CLR     @#TEMP6             ;0 1 SHIFTED BY 0=0 1, PS=0

027724  010502                    REG01:  MOV     R5,R2               ;SAVE R5
027726  013700  025416                    MOV     @#TEMP1,%0          ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 0
027732  013701  025420                    MOV     @#TEMP2,%0!1        ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 1
027736  000241                            CLC
027740  032737  000001  001006            BIT     #1,@#$PASS          ;IS IT AN EVEN PASS ?
027746  001004                            BNE     2$                  ;IF NOT THEN GO TO 2$
027750  013705  025422                    MOV     @#TEMP3,R5          ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
027754  073005                            ASHC    R5,R0               ;USING R0
027756  000402                            BR      4$
027760  073067  175436            2$:     ASHC    TEMP3,%0            ;ASHC REGISTER 0 BY THE CONTENTS OF TEMP3
027764  106737  025414            4$:     MFPS    @#PSWORD            ;SAVE PS
027770  123737  025430  025414            CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
027776  001401                            BEQ     11$
030000  104000                            EMT                         ;WRONG PS
030002  005237  025412            11$:    INC     @#COUNT
030006  023700  025424                    CMP     @#TEMP4,%0          ;IS THE RESULT IN R0 SAME AS TEMP4?
030012  001401                            BEQ     12$
030014  104000                            EMT                         ;WRONG RESULT IN R0
030016  023701  025426            12$:    CMP     @#TEMP5,%1          ;IS THE RESULT IN R1 SAME AS TEMP5?
                                                                      ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
                                                                      ;AND PS=TEMP6
030022  001401                            BEQ     13$
030024  104000                            EMT                         ;WRONG RESULT IN R1
030026  010205                    13$:    MOV     R2,R5               ;RESTORE R5
030030  021537  025412                    CMP     (R5),@#COUNT        ;IS TEST NUMBER=COUNTER?
030034  001401                            BEQ     14$
030036  104000                            EMT                         ;NO
030040  005215                    14$:    INC     (R5)
030042  021527  000160                    CMP     (R5),#160           ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
030046  002014                            BGE     6$                  ;YES
030050  005237  025422                    INC     @#TEMP3
030054  000241                            CLC
030056  006137  025426                    ROL     @#TEMP5             ;ROTATE TEMP5 LEFT BY 1 PLACE
030062  006137  025424                    ROL     @#TEMP4             ;INTRODUCE CARRY FROM TEMP4 IN TEMP5
030066  021527  000121                    CMP     (R5),#121           ;IS IT TEST 121?
030072  001004                            BNE     REGR23
```

```
030074  004467  000344                    JSR     R4,RITSH             ;IF SO THEN GO AND INITIATE RIGHT SHIFT
030100  004767  000374          6$:       JSR     %7,TST160
030104  013702  025416          REGR23:   MOV     @#TEMP1,%2           ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 2
030110  013703  025420                    MOV     @#TEMP2,%2!1         ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 3
030114  000241                            CLC
030116  032737  000001  001006            BIT     #1,@#$PASS           ;IS IT AN EVEN PASS ?
030124  001004                            BNE     2$                   ;IF NOT THEN GO TO 2$
030126  013704  025422                    MOV     @#TEMP3,R4           ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
030132  073204                            ASHC    R4,R2                ;USING R2
030134  000402                            BR      4$
030136  073267  175260          2$:       ASHC    TEMP3,%2             ;ASHC REGISTER 2 BY THE CONTETNS OF TEMP3
030142  106737  025414          4$:       MFPS    @#PSWORD             ;SAVE PS
030146  123737  025430  025414            CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
030154  001401                            BEQ     11$
030156  104000                            EMT                          ;WRONG PS
030160  005237  025412          11$:      INC     @#COUNT
030164  023702  025424                    CMP     @#TEMP4,%2           ;IS THE RESULT IN R2 SAME AS TEMP4?
030170  001401                            BEQ     12$
030172  104000                            EMT                          ;WRONG RESULT IN R2
030174  023703  025426          12$:      CMP     @#TEMP5,%3           ;IS THE RESULT IN R3 SAME AS TEMP5?
                                                                       ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
                                                                       ;AND PS=TEMP6
030200  001401                            BEQ     13$
030202  104000                            EMT                          ;WRONG RESULT IN R1
030204  021537  025412          13$:      CMP     (R5),@#COUNT         ;IS TEST NUMBER=COUNTER?
030210  001401                            BEQ     14$
030212  104000                            EMT                          ;NO
030214  005215                  14$:      INC     (R5)
030216  021527  000160                    CMP     (R5),#160            ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
030222  002014                            BGE     6$                   ;YES
030224  005237  025422                    INC     @#TEMP3
030230  000241                            CLC
030232  006137  025426                    ROL     @#TEMP5              ;ROTATE TEMP5 LEFT BY 1 PLACE
030236  006137  025424                    ROL     @#TEMP4              ;INTRODUCE CARRY FROM TEMP5 IN TEMP4
030242  021527  000121                    CMP     (R5),#121            ;IS IT TEST 121?
030246  001004                            BNE     REG45
030250  004467  000170                    JSR     R4,RITSH             ;IF SO THEN GO AND INITIATE RIGHT SHIFT
030254  004767  000220          6$:       JSR     %7,TST160
030260  010501                  REG45:    MOV     R5,R1                ;SAVE R5
030262  013704  025416                    MOV     @#TEMP1,%4           ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 4
030266  013705  025420                    MOV     @#TEMP2,%4!1         ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 5
030272  000241                            CLC
030274  032737  000001  001006            BIT     #1,@#$PASS           ;IS IT AN EVEN PASS ?
030302  001004                            BNE     2$                   ;IF NOT THEN GO TO 2$
030304  013700  025422                    MOV     @#TEMP3,R0           ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
030310  073400                            ASHC    R0,R4                ;USING R4
030312  000402                            BR      4$
030314  073467  175102          2$:       ASHC    TEMP3,%4             ;ASHC REGISTER 4 BY THE CONTENTS OF TEMP3
030320  106737  025414          4$:       MFPS    @#PSWORD             ;SAVE PS
030324  123737  025430  025414            CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
030332  001401                            BEQ     11$
030334  104000                            EMT                          ;WRONG PS
030336  005237  025412          11$:      INC     @#COUNT
030342  023704  025424                    CMP     @#TEMP4,%4           ;IS THE RESULT IN R4 SAME AS TEMP4?
030346  001401                            BEQ     12$
030350  104000                            EMT                          ;WRONG RESULT IN R4
```

```
030352  023705  025426          12$:    CMP     @#TEMP5,%5          ;IS THE RESULT IN R5 SAME AS TEMP5?
                                                                    ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
                                                                    ;AND PS=TEMP6
030356  001401                          BEQ     13$
030360  104000                          EMT                         ;WRONG RESULT IN R5
030362  021137  025412          13$:    CMP     (R1),@#COUNT        ;IS TEST NUMBER=COUNTER?
030366  001401                          BEQ     14$
030370  104000                          EMT                         ;NO
030372  010105                  14$:    MOV     R1,R5               ;RESTORE R5
030374  005215                          INC     (R5)
030376  021527  000160                  CMP     (R5),#160           ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
030402  002014                          BGE     6$                  ;YES
030404  005237  025422                  INC     @#TEMP3
030410  000241                          CLC
030412  006137  025426                  ROL     @#TEMP5             ;ROTATE TEMP5 LEFT BY 1 PLACE
030416  006137  025424                  ROL     @#TEMP4             ;INTRODUCE CARRY FROM TEMP5 IN TEMP4
030422  021527  000121                  CMP     (R5),#121           ;IS IT TEST 121?
030426  001004                          BNE     8$
030430  004467  000010                  JSR     R4,RITSH            ;IF SO THEN GO AND INITIATE RIGHT SHIFT
030434  004767  000040          6$:     JSR     %7,TST160
030440  000167  177260          8$:     JMP     REG01
030444  022424                  RITSH:  CMP     (R4)+,(R4)+         ;MAKE R4 POINT TO THE NEXT REG TAG
030446  012737  040000  025416          MOV     #40000,@#TEMP1      ;TEMP1=4000
030454  005037  025420                  CLR     @#TEMP2             ;TEMP2=0
030460  012737  177742  025422          MOV     #-30.,@#TEMP3       ;TEMP3=-30
030466  005037  025424                  CLR     @#TEMP4             ;TEMP4=0
030472  005237  025426                  INC     @#TEMP5             ;TEMP5=1
030476  000204                          RTS     R4
030500  021527  000160          TST160: CMP     (R5),#160           ;IS IT TEST 160
030504  001010                          BNE     TST161              ;IF NOT THEN TRY TEST 161
030506  005037  025416                  CLR     @#TEMP1             ;0 0 SHIFTED BY 0
030512  005037  025424                  CLR     @#TEMP4             ;IS EQUAL TO 0 0
030516  012737  000004  025430          MOV     #4,@#TEMP6          ;AND PS=4
030524  000207                          RTS     %7
030526  021527  000161          TST161: CMP     (R5),#161           ;IS IT TEST 161
030532  001004                          BNE     TST162
030534  012737  177746  025422          MOV     #-32,@#TEMP3        ;0 0 SHIFTED BY -32=0 0, PS=4
030542  000207                          RTS     %7
030544  021527  000162          TST162: CMP     (R5),#162           ;IS IT TEST 162
030550  001004                          BNE     TST163              ;IF NOT THEN TRY TEST 163
030552  012737  000032  025422          MOV     #32,@#TEMP3         ;0 0 SHIFTED BY 32=0 0, PS=4
030560  000207                          RTS     %7
030562  021527  000163          TST163: CMP     (R5),#163           ;IS IT TEST 163?
030566  001016                          BNE     TST164              ;IF NOT THEN TRY TEST 164
030570  012737  052525  025416          MOV     #52525,@#TEMP1      ;52525 0
030576  012737  177760  025422          MOV     #-16.,@#TEMP3       ;SHIFTED BY -16.
030604  005037  025424                  CLR     @#TEMP4
030610  012737  052525  025426          MOV     #52525,@#TEMP5      ;IS EQUAL TO 0 52525
030616  005037  025430                  CLR     @#TEMP6             ;AND PS = 0
030622  000207                          RTS     %7
030624  021527  000164          TST164: CMP     (R5),#164           ;IS IT TEST 164?
030630  001014                          BNE     TST165              ;IF NOT THEN TRY TEST 165
030632  012737  125252  025416          MOV     #125252,@#TEMP1     ;125252 0 SHIFTED BY -16.
030640  005337  025424                  DEC     @#TEMP4
030644  012737  125252  025426          MOV     #125252,@#TEMP5     ;IS EQUAL TO -1 125252
030652  012737  000010  025430          MOV     #10,@#TEMP6         ;AND PS=10
```

```
030660  000207                          RTS     %7
030662  021527  000165          TST165: CMP     (R5),#165       ;IS IT TEST 165?
030666  001007                          BNE     TST166          ;IF NOT THEN TRY TEST 166
030670  012737  177777  025416          MOV     #-1,@#TEMP1     ;-1 0 SHIFTED BY -16
030676  012737  177777  025426          MOV     #-1,@#TEMP5     ;IS EQUAL TO -1 -1, AND PS=10
030704  000207                          RTS     %7
030706  021527  000166          TST166: CMP     (R5),#166       ;IS IT TEST 166?
030712  001011                          BNE     TST167          ;IF NOT THEN TRY TEST 167
030714  012737  100000  025416          MOV     #100000,@#TEMP1 ;100000 0
030722  012737  177740  025422          MOV     #-32.,@#TEMP3   ;SHIFTED BY -32 IS EQUAL TO -1 -1
030730  005237  025430                  INC     @#TEMP6         ;AND PS=11
030734  000207                          RTS     %7
030736  021527  000167          TST167: CMP     (R5),#167       ;IS IT TEST 167?
030742  001014                          BNE     TST170          ;IF NOT THEN TRY TEST 170
```

```
 7617  030744  005037  025416          CLR     @#TEMP1
 7618  030750  005337  025420          DEC     @#TEMP2          ;0 -1
```

```
 7619  030754  012737  000020  025422           MOV     #16.,@#TEMP3    ;SHIFTED BY 16.
 7620  030762  005037  025426                   CLR     @#TEMP5         ;IS EQUAL TO -1 0
 7621  030766  005237  025430                   INC     @#TEMP6         ;AND PS=12
 7622  030772  000207                           RTS     %7
 7623  030774  021527  000170           TST170: CMP     (R5),#170       ;IS IT TEST 170?
```

```
7624  031000  001007          BNE     TST171          ;IF NOT THEN TRY TEST 171
```

```
031002  012737  125252  025420           MOV   #125252,@#TEMP2 ;0 125252 SHIFTED BY 16
031010  012737  125252  025424           MOV   #125252,@#TEMP4 ;IS EQUAL TO 125252 0, AND PS=12
031016  000207                           RTS   %7
031020  021527  000171          TST171:  CMP   (R5),#171       ;IS IT TEST 171?
031024  001010                           BNE   TST172          ;IF NOT THEN TRY TEST 172
031026  005337  025422                   DEC   @#TEMP3         ;0 125252 SHIFTED BY 15
031032  012737  052525  025424           MOV   #52525,@#TEMP4  ;IS EQUAL TO 52525 0
031040  005037  025430                   CLR   @#TEMP6         ;AND PS=0
031044  000207                           RTS   %7
031046  021527  000172          TST172:  CMP   (R5),#172       ;IS IT TEST 172?
031052  001006                           BNE   TST173          ;IF NOT THEN TRY TEST 173
031054  012737  052525  025420           MOV   #52525,@#TEMP2  ;0 52525
031062  005237  025422                   INC   @#TEMP3         ;SHIFTED BY 16. IS EQUAL TO 52525 0, AND PS=0
031066  000207                           RTS   %7
031070  021527  000173          TST173:  CMP   (R5),#173       ;IS IT TEST 173?
031074  001014                           BNE   TST174          ;IF NOT THEN TRY TEST 174
031076  012737  177777  025420           MOV   #-1,@#TEMP2     ;0 -1
031104  005337  025422                   DEC   @#TEMP3         ;SHIFTED BY 15.
031110  012737  077777  025424           MOV   #77777,@#TEMP4
031116  012737  100000  025426           MOV   #100000,@#TEMP5 ;IS EQUAL TO 77777 100000, AND PS=0
031124  000207                           RTS   %7
031126  021527  000174          TST174:  CMP   (R5),#174       ;IS IT TEST 174?
031132  001013                           BNE   TST175          ;IF NOT THEN TRY TEST 175
031134  012737  100000  025416           MOV   #100000,@#TEMP1
031142  005337  025420                   DEC   @#TEMP2         ;100000 -2 SHIFTED BY 15.
031146  005037  025426                   CLR   @#TEMP5         ;IS EQUAL TO 77777 0
031152  012737  000002  025430           MOV   #2,@#TEMP6      ;AND PS=2
031160  000207                           RTS   %7
031162  021527  000175          TST175:  CMP   (R5),#175       ;IS IT TEST 175?
031166  001015                           BNE   ENT176          ;IF NOT THEN TRY TEST 176
031170  012737  177777  025416           MOV   #-1,@#TEMP1
031176  005037  025420                   CLR   @#TEMP2         ;-1 0
031202  005237  025422                   INC   @#TEMP3         ;SHIFTED BY 16.
031206  005037  025424                   CLR   @#TEMP4         ;IS EQUAL TO 0 0
031212  012737  000007  025430           MOV   #7,@#TEMP6      ;AND PS=7
```

```
031220  000207                          RTS     %7
031222  021527  000176          ENT176: CMP     (R5),#176       ;IS THE PROGRM ENTERING TEST 176?
031226  001401                          BEQ     1$
031230  104000                          EMT                     ;TEST NUMBER GOOFED

031232  005726                  1$:     TST     (SP)+           ;RESTORE STACK POINTER

                                ;*******************************************************************
                                ;TEST:176       1 SHIFTED BY 8. = 400  PS = 0
                                ;*******************************************************************

031234                          TST176:
031234  012701  000000                  MOV     #DUMMY,%1       ;LOAD R1 WITH DUMMY
031240  012701  000001                  MOV     #1,%1!1         ;LOAD R1!1 WITH 1
031244  000241                          CLC
031246  073127  000010                  ASHC    #8.,%1          ;SHIFT R1,R1!1 BY 8.
031252  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
031256  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS THE PS 0?
031264  001401                          BEQ     11$
031266  104000                          EMT                     ;THE PS IS NOT EQUAL TO 0
031270  022701  000400          11$:    CMP     #400,%1         ;IS THE RESULT 400?
031274  001401                          BEQ     13$
031276  104000                          EMT                     ;R1 IS NOT EQUAL TO 400
031300                          13$:
031300  005215                          INC     (R5)


                                ;*******************************************************************
                                ;TEST:177       -1 SHIFTED BY 15. = 100000  PS = 11
                                ;*******************************************************************

031302                          TST177:
031302  012703  000000                  MOV     #DUMMY,%3       ;LOAD R3 WITH DUMMY
031306  012703  177777                  MOV     #-1,%3!1        ;LOAD R3!1 WITH -1
031312  000241                          CLC
031314  073327  000017                  ASHC    #15.,%3         ;SHIFT R3,R3!1 BY 15.
031320  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
031324  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS THE PS 11?
031332  001401                          BEQ     11$
031334  104000                          EMT                     ;THE PS IS NOT EQUAL TO 11
031336  022703  100000          11$:    CMP     #100000,%3      ;IS THE RESULT 100000?
031342  001401                          BEQ     13$
031344  104000                          EMT                     ;R3 IS NOT EQUAL TO 100000
031346                          13$:
031346  005215                          INC     (R5)


                                ;*******************************************************************
                                ;TEST:200       52525 SHIFTED BY 0 = 52525  PS = 0
                                ;*******************************************************************

031350                          TST200:
031350  010501                          MOV     R5,R1           ;SAVE R5
031352  012705  000000                  MOV     #DUMMY,%5       ;LOAD R5 WITH DUMMY
031356  012705  052525                  MOV     #52525,%5!1     ;LOAD R5!1 WITH 52525
031362  000241                          CLC
```

```
031364 073527 000000                  ASHC    #0,%5            ;SHIFT R5,R5!1 BY 0
031370 106737 025414                  MFPS    @#PSWORD         ;SAVE PS
031374 122737 000000 025414           CMPB    #0,@#PSWORD      ;IS THE PS 0?
031402 001401                         BEQ     11$
031404 104000                         EMT                      ;THE PS IS NOT EQUAL TO 0
031406 022705 052525          11$:    CMP     #52525,%5        ;IS THE RESULT 52525?
031412 001401                         BEQ     13$
031414 104000                         EMT                      ;R5 IS NOT EQUAL TO 52525
031416                        13$:
031416 010105                         MOV     R1,R5            ;RESTORE R5
031420 005215                         INC     (R5)

                              ;*******************************************************************
                              ;TEST:201        20010 SHIFTED BY -13. = 101  PS = 0
                              ;*******************************************************************

031422                        TST201:
031422 012701 000000                  MOV     #DUMMY,%1        ;LOAD R1 WITH DUMMY
031426 012701 020010                  MOV     #20010,%1!1      ;LOAD R1!1 WITH 20010
031432 000241                         CLC
031434 073127 177763                  ASHC    #-13.,%1         ;SHIFT R1,R1!1 BY -13.
031440 106737 025414                  MFPS    @#PSWORD         ;SAVE PS
031444 122737 000000 025414           CMPB    #0,@#PSWORD      ;IS THE PS 0?
031452 001401                         BEQ     11$
031454 104000                         EMT                      ;THE PS IS NOT EQUAL TO 0
031456 022701 000101          11$:    CMP     #101,%1          ;IS THE RESULT 101?
031462 001401                         BEQ     13$
031464 104000                         EMT                      ;R1 IS NOT EQUAL TO 101
031466                        13$:
031466 005215                         INC     (R5)

                              ;*******************************************************************
                              ;TEST:202        -1 SHIFTED BY 16. = 0  PS = 11
                              ;*******************************************************************

031470                        TST202:
031470 012703 000000                  MOV     #DUMMY,%3        ;LOAD R3 WITH DUMMY
031474 012703 177777                  MOV     #-1,%3!1         ;LOAD R3!1 WITH -1
031500 000241                         CLC
031502 073327 000020                  ASHC    #16.,%3          ;SHIFT R3,R3!1 BY 16.
031506 106737 025414                  MFPS    @#PSWORD         ;SAVE PS
031512 122737 000011 025414           CMPB    #11,@#PSWORD     ;IS THE PS 11?
031520 001401                         BEQ     11$
031522 104000                         EMT                      ;THE PS IS NOT EQUAL TO 11
031524 022703 000000          11$:    CMP     #0,%3            ;IS THE RESULT 0?
031530 001401                         BEQ     13$
031532 104000                         EMT                      ;R3 IS NOT EQUAL TO 0
031534                        13$:
031534 005215                         INC     (R5)

                              ;*******************************************************************
                              ;TEST:203        1 SHIFTED BY -1 = 100000  PS = 1
                              ;*******************************************************************
```

```

031536                                  TST203:
031536  010501                                  MOV     R5,R1           ;SAVE R5
031540  012705  000000                          MOV     #DUMMY,%5       ;LOAD R5 WITH DUMMY
031544  012705  000001                          MOV     #1,%5!1         ;LOAD R5!1 WITH 1
031550  000241                                  CLC
031552  073527  177777                          ASHC    #-1,%5          ;SHIFT R5,R5!1 BY -1
031556  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
031562  122737  000001  025414                  CMPB    #1,@#PSWORD     ;IS THE PS 1?
031570  001401                                  BEQ     11$
031572  104000                                  EMT                     ;THE PS IS NOT EQUAL TO 1
031574  022705  100000                  11$:    CMP     #100000,%5      ;IS THE RESULT 100000?
031600  001401                                  BEQ     13$
031602  104000                                  EMT                     ;R5 IS NOT EQUAL TO 100000
031604                                  13$:
031604  010105                                  MOV     R1,R5           ;RESTORE R5
031606  005215                                  INC     (R5)


                                        ;*************************************************************************
                                        ;TEST:204       125252 SHIFTED BY -16. = 125252  PS = 11
                                        ;*************************************************************************

031610                                  TST204:
031610  012701  000000                          MOV     #DUMMY,%1       ;LOAD R1 WITH DUMMY
031614  012701  125252                          MOV     #125252,%1!1    ;LOAD R1!1 WITH 125252
031620  000241                                  CLC
031622  073127  177760                          ASHC    #-16.,%1        ;SHIFT R1,R1!1 BY -16.
031626  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
031632  122737  000011  025414                  CMPB    #11,@#PSWORD    ;IS THE PS 11?
031640  001401                                  BEQ     11$
031642  104000                                  EMT                     ;THE PS IS NOT EQUAL TO 11
031644  022701  125252                  11$:    CMP     #125252,%1      ;IS THE RESULT 125252?
031650  001401                                  BEQ     13$
031652  104000                                  EMT                     ;R1 IS NOT EQUAL TO 125252
031654                                  13$:
031654  005215                                  INC     (R5)


                                        ;*************************************************************************
                                        ;TEST:205       125252 125252 SHIFTED BY 21. = 52500 000000  PS = 3
                                        ;*************************************************************************

031656                                  TST205:
031656  012702  125252                          MOV     #125252,%2      ;LOAD R2 WITH 125252
031662  012703  125252                          MOV     #125252,%2!1    ;LOAD R2!1 WITH 125252
031666  000241                                  CLC
031670  073227  000025                          ASHC    #21.,%2         ;SHIFT R2,R2!1 BY 21.
031674  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
031700  122737  000003  025414                  CMPB    #3,@#PSWORD     ;IS THE PS 3?
031706  001401                                  BEQ     11$
031710  104000                                  EMT                     ;THE PS IS NOT EQUAL TO 3
031712  022702  052500                  11$:    CMP     #52500,%2       ;IS THE RESULT 52500?
031716  001401                                  BEQ     12$
031720  104000                                  EMT                     ;R2 IS NOT EQUAL TO 52500
031722  022703  000000                  12$:    CMP     #000000,%2!1    ;IS THE RESULT 000000?
```

```
031726  001401                          BEQ     13$
031730  104000                          EMT                             ;R2!1 IS NOT EQUAL TO 000000
031732                          13$:
031732  005215                          INC     (R5)



031734  012702  177771                  MOV     #-7,%2
031740  012703  025432                  MOV     #S1,%3
031744  012704  025434                  MOV     #S2,%4

                                ;*****************************************************************
                                ;TEST:206       125252 125252 SHIFTED BY S1 = 177525 52525  PS = 10
                                ;*****************************************************************

031750                          TST206:
031750  012700  125252                  MOV     #125252,%0                  ;LOAD R0 WITH 125252
031754  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
031760  000241                          CLC
031762  073067  173444                  ASHC    S1,%0           ;SHIFT R0,R0!1 BY S1
031766  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
031772  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
032000  001401                          BEQ     11$
032002  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
032004  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032010  001401                          BEQ     12$
032012  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
032014  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032020  001401                          BEQ     13$
032022  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032024                          13$:
032024  005215                          INC     (R5)


                                ;*****************************************************************
                                ;TEST:207       125252 125252 SHIFTED BY @S2 = 177525 52525  PS = 10
                                ;*****************************************************************

032026                          TST207:
032026  012700  125252                  MOV     #125252,%0                  ;LOAD R0 WITH 125252
032032  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032036  000241                          CLC
032040  073077  173370                  ASHC    @S2,%0          ;SHIFT R0,R0!1 BY @S2
032044  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032050  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
032056  001401                          BEQ     11$
032060  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
032062  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032066  001401                          BEQ     12$
032070  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
032072  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032076  001401                          BEQ     13$
032100  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032102                          13$:
032102  005215                          INC     (R5)
```

```
                                        ;*******************************************************************
                                        ;TEST:210       125252 125252 SHIFTED BY @#S1 = 177525 52525  PS = 10
                                        ;*******************************************************************

032104                                  TST210:
032104  012700  125252                          MOV     #125252,%0              ;LOAD R0 WITH 125252
032110  012701  125252                          MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032114  000241                                  CLC
032116  073037  025432                          ASHC    @#S1,%0         ;SHIFT R0,R0!1 BY @#S1
032122  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
032126  122737  000010  025414                  CMPB    #10,@#PSWORD    ;IS THE PS 10?
032134  001401                                  BEQ     11$
032136  104000                                  EMT                     ;THE PS IS NOT EQUAL TO 10
032140  022700  177525                  11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032144  001401                                  BEQ     12$
032146  104000                                  EMT                     ;R0 IS NOT EQUAL TO 177525
032150  022701  052525                  12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032154  001401                                  BEQ     13$
032156  104000                                  EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032160                                  13$:
032160  005215                                  INC     (R5)


                                        ;*******************************************************************
                                        ;TEST:211       125252 125252 SHIFTED BY (3) = 177525 52525  PS = 10
                                        ;*******************************************************************

032162                                  TST211:
032162  012700  125252                          MOV     #125252,%0              ;LOAD R0 WITH 125252
032166  012701  125252                          MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032172  000241                                  CLC
032174  073013                                  ASHC    (3),%0          ;SHIFT R0,R0!1 BY (3)
032176  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
032202  122737  000010  025414                  CMPB    #10,@#PSWORD    ;IS THE PS 10?
032210  001401                                  BEQ     11$
032212  104000                                  EMT                     ;THE PS IS NOT EQUAL TO 10
032214  022700  177525                  11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032220  001401                                  BEQ     12$
032222  104000                                  EMT                     ;R0 IS NOT EQUAL TO 177525
032224  022701  052525                  12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032230  001401                                  BEQ     13$
032232  104000                                  EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032234                                  13$:
032234  005215                                  INC     (R5)


                                        ;*******************************************************************
                                        ;TEST:212       125252 125252 SHIFTED BY (3)+ = 177525 52525  PS = 10
                                        ;*******************************************************************

032236                                  TST212:
032236  012700  125252                          MOV     #125252,%0              ;LOAD R0 WITH 125252
032242  012701  125252                          MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032246  000241                                  CLC
032250  073023                                  ASHC    (3)+,%0         ;SHIFT R0,R0!1 BY (3)+
```

```
032252  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032256  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
032264  001401                          BEQ     11$
032266  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
032270  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032274  001401                          BEQ     12$
032276  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
032300  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032304  001401                          BEQ     13$
032306  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032310                          13$:
032310  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:213       125252 125252 SHIFTED BY -(3) = 177525 52525  PS = 10
                                ;*************************************************************************

032312                          TST213:
032312  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
032316  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032322  000241                          CLC
032324  073043                          ASHC    -(3),%0         ;SHIFT R0,R0!1 BY -(3)
032326  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032332  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
032340  001401                          BEQ     11$
032342  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
032344  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032350  001401                          BEQ     12$
032352  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
032354  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032360  001401                          BEQ     13$
032362  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032364                          13$:
032364  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:214       125252 125252 SHIFTED BY 2(4) = 177252 125252  PS = 11
                                ;*************************************************************************

032366                          TST214:
032366  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
032372  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032376  000241                          CLC
032400  073064  000002                  ASHC    2(4),%0         ;SHIFT R0,R0!1 BY 2(4)
032404  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032410  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS THE PS 11?
032416  001401                          BEQ     11$
032420  104000                          EMT                     ;THE PS IS NOT EQUAL TO 11
032422  022700  177252          11$:    CMP     #177252,%0      ;IS THE RESULT 177252?
032426  001401                          BEQ     12$
032430  104000                          EMT                     ;R0 IS NOT EQUAL TO 177252
032432  022701  125252          12$:    CMP     #125252,%0!1    ;IS THE RESULT 125252?
032436  001401                          BEQ     13$
032440  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 125252 OR INCORRECT SEQUENCE
```

```
032442                            13$:
032442  005215                            INC     (R5)


                                  ;*********************************************************************
                                  ;TEST:215          125252 125252 SHIFTED BY @(4) = 177525 52525  PS = 10
                                  ;*********************************************************************

032444                            TST215:
032444  012700  125252                    MOV     #125252,%0              ;LOAD R0 WITH 125252
032450  012701  125252                    MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032454  000241                            CLC
032456  073074  000000                    ASHC    @(4),%0         ;SHIFT R0,R0!1 BY @(4)
032462  106737  025414                    MFPS    @#PSWORD        ;SAVE PS
032466  122737  000010  025414            CMPB    #10,@#PSWORD    ;IS THE PS 10?
032474  001401                            BEQ     11$
032476  104000                            EMT                     ;THE PS IS NOT EQUAL TO 10
032500  022700  177525            11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032504  001401                            BEQ     12$
032506  104000                            EMT                     ;R0 IS NOT EQUAL TO 177525
032510  022701  052525            12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032514  001401                            BEQ     13$
032516  104000                            EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032520                            13$:
032520  005215                            INC     (R5)


                                  ;*********************************************************************
                                  ;TEST:216          125252 125252 SHIFTED BY @(4)+ = 177525 52525  PS = 10
                                  ;*********************************************************************

032522                            TST216:
032522  012700  125252                    MOV     #125252,%0              ;LOAD R0 WITH 125252
032526  012701  125252                    MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032532  000241                            CLC
032534  073034                            ASHC    @(4)+,%0                ;SHIFT R0,R0!1 BY @(4)+
032536  106737  025414                    MFPS    @#PSWORD        ;SAVE PS
032542  122737  000010  025414            CMPB    #10,@#PSWORD    ;IS THE PS 10?
032550  001401                            BEQ     11$
032552  104000                            EMT                     ;THE PS IS NOT EQUAL TO 10
032554  022700  177525            11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032560  001401                            BEQ     12$
032562  104000                            EMT                     ;R0 IS NOT EQUAL TO 177525
032564  022701  052525            12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032570  001401                            BEQ     13$
032572  104000                            EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032574                            13$:
032574  005215                            INC     (R5)


                                  ;*********************************************************************
                                  ;TEST:217          125252 125252 SHIFTED BY @-(4) = 177525 52525  PS = 10
                                  ;*********************************************************************

032576                            TST217:
032576  012700  125252                    MOV     #125252,%0              ;LOAD R0 WITH 125252
```

```
032602  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
032606  000241                          CLC
032610  073054                          ASHC    @-(4),%0                ;SHIFT R0,R0!1 BY @-(4)
032612  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032616  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS THE PS 10?
032624  001401                          BEQ     11$
032626  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
032630  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
032634  001401                          BEQ     12$
032636  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
032640  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
032644  001401                          BEQ     13$
032646  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
032650                          13$:
032650  005215                          INC     (R5)
```

```
                                        ;*********************************
                                        ;     MUL INSTRUCTION TESTS
                                        ;*********************************




                                ;*******************************************************************
                                ;TEST:220       MUL       1 * #0 = 0 0        PS = 4
                                ;*******************************************************************

032652                          TST220:
032652  012700  000001                  MOV     #1,%0           ;LOAD MULTIPLICAND WITH 1
032656  070027  000000                  MUL     #0,%0           ;MULTIPLY 1 * #0
032662  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032666  122737  000004  025414          CMPB    #4,@#PSWORD     ;IS PS = 4
032674  001401                          BEQ     11$
032676  104000                          EMT                     ;PS IS WRONG
032700  022700  000000          11$:    CMP     #0,%0           ;IS HIGH ORDER = 0
032704  001401                          BEQ     12$
032706  104000                          EMT                     ;HIGH ORDER IS WRONG
032710  022701  000000          12$:    CMP     #0,%0!1         ;IS LOW ORDER = 0
032714  001401                          BEQ     13$
032716  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
032720                          13$:
032720  005215                          INC     (R5)


                                ;*******************************************************************
                                ;TEST:221       MUL       -1 * #1 = -1 -1        PS = 10
                                ;*******************************************************************

032722                          TST221:
032722  012700  177777                  MOV     #-1,%0          ;LOAD MULTIPLICAND WITH -1
032726  070027  000001                  MUL     #1,%0           ;MULTIPLY -1 * #1
032732  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
032736  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS PS = 10
032744  001401                          BEQ     11$
032746  104000                          EMT                     ;PS IS WRONG
032750  022700  177777          11$:    CMP     #-1,%0          ;IS HIGH ORDER = -1
032754  001401                          BEQ     12$
032756  104000                          EMT                     ;HIGH ORDER IS WRONG
032760  022701  177777          12$:    CMP     #-1,%0!1        ;IS LOW ORDER = -1
032764  001401                          BEQ     13$
032766  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
032770                          13$:
032770  005215                          INC     (R5)


                                ;*******************************************************************
                                ;TEST:222       MUL       2 * #2 = 0 4        PS = 0
                                ;*******************************************************************

032772                          TST222:
032772  012702  000002                  MOV     #2,%2           ;LOAD MULTIPLICAND WITH 2
032776  070227  000002                  MUL     #2,%2           ;MULTIPLY 2 * #2
```

```
033002 106737 025414                   MFPS    @#PSWORD          ;SAVE PS
033006 122737 000000 025414            CMPB    #0,@#PSWORD       ;IS PS = 0
033014 001401                          BEQ     11$
033016 104000                          EMT                       ;PS IS WRONG
033020 022702 000000           11$:    CMP     #0,%2             ;IS HIGH ORDER = 0
033024 001401                          BEQ     12$
033026 104000                          EMT                       ;HIGH ORDER IS WRONG
033030 022703 000004           12$:    CMP     #4,%2!1           ;IS LOW ORDER = 4
033034 001401                          BEQ     13$
033036 104000                          EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033040                         13$:
033040 005215                          INC     (R5)


                               ;*******************************************************************
                               ;TEST:223        MUL      1000 * #200 = 1 0         PS = 1
                               ;*******************************************************************

033042                         TST223:
033042 010501                          MOV     R5,R1             ;SAVE R5
033044 012704 001000                   MOV     #1000,%4          ;LOAD MULTIPLICAND WITH 1000
033050 070427 000200                   MUL     #200,%4           ;MULTIPLY 1000 * #200
033054 106737 025414                   MFPS    @#PSWORD          ;SAVE PS
033060 122737 000001 025414            CMPB    #1,@#PSWORD       ;IS PS = 1
033066 001401                          BEQ     11$
033070 104000                          EMT                       ;PS IS WRONG
033072 022704 000001           11$:    CMP     #1,%4             ;IS HIGH ORDER = 1
033076 001401                          BEQ     12$
033100 104000                          EMT                       ;HIGH ORDER IS WRONG
033102 022705 000000           12$:    CMP     #0,%4!1           ;IS LOW ORDER = 0
033106 001401                          BEQ     13$
033110 104000                          EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033112                         13$:
033112 010105                          MOV     R1,R5             ;RESTORE R5
033114 005215                          INC     (R5)


                               ;*******************************************************************
                               ;TEST:224        MUL      2 * #77777 = 0 177776         PS = 1
                               ;*******************************************************************

033116                         TST224:
033116 012700 000002                   MOV     #2,%0             ;LOAD MULTIPLICAND WITH 2
033122 070027 077777                   MUL     #77777,%0         ;MULTIPLY 2 * #77777
033126 106737 025414                   MFPS    @#PSWORD          ;SAVE PS
033132 122737 000001 025414            CMPB    #1,@#PSWORD       ;IS PS = 1
033140 001401                          BEQ     11$
033142 104000                          EMT                       ;PS IS WRONG
033144 022700 000000           11$:    CMP     #0,%0             ;IS HIGH ORDER = 0
033150 001401                          BEQ     12$
033152 104000                          EMT                       ;HIGH ORDER IS WRONG
033154 022701 177776           12$:    CMP     #177776,%0!1      ;IS LOW ORDER = 177776
033160 001401                          BEQ     13$
033162 104000                          EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033164                         13$:
033164 005215                          INC     (R5)
```

```


                                    ;*******************************************************************
                                    ;TEST:225       MUL       7777 * #10 = 0 77770        PS = 0
                                    ;*******************************************************************

033166                              TST225:
033166  012702  007777                      MOV     #7777,%2           ;LOAD MULTIPLICAND WITH 7777
033172  070227  000010                      MUL     #10,%2             ;MULTIPLY 7777 * #10
033176  106737  025414                      MFPS    @#PSWORD           ;SAVE PS
033202  122737  000000  025414              CMPB    #0,@#PSWORD        ;IS PS = 0
033210  001401                              BEQ     11$
033212  104000                              EMT                        ;PS IS WRONG
033214  022702  000000              11$:    CMP     #0,%2              ;IS HIGH ORDER = 0
033220  001401                              BEQ     12$
033222  104000                              EMT                        ;HIGH ORDER IS WRONG
033224  022703  077770              12$:    CMP     #77770,%2!1        ;IS LOW ORDER = 77770
033230  001401                              BEQ     13$
033232  104000                              EMT                        ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033234                              13$:
033234  005215                              INC     (R5)


                                    ;*******************************************************************
                                    ;TEST:226       MUL       77777 * #77777 = 37777 1        PS = 1
                                    ;*******************************************************************

033236                              TST226:
033236  010501                              MOV     R5,R1              ;SAVE R5
033240  012704  077777                      MOV     #77777,%4          ;LOAD MULTIPLICAND WITH 77777
033244  070427  077777                      MUL     #77777,%4          ;MULTIPLY 77777 * #77777
033250  106737  025414                      MFPS    @#PSWORD           ;SAVE PS
033254  122737  000001  025414              CMPB    #1,@#PSWORD        ;IS PS = 1
033262  001401                              BEQ     11$
033264  104000                              EMT                        ;PS IS WRONG
033266  022704  037777              11$:    CMP     #37777,%4          ;IS HIGH ORDER = 37777
033272  001401                              BEQ     12$
033274  104000                              EMT                        ;HIGH ORDER IS WRONG
033276  022705  000001              12$:    CMP     #1,%4!1            ;IS LOW ORDER = 1
033302  001401                              BEQ     13$
033304  104000                              EMT                        ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033306                              13$:
033306  010105                              MOV     R1,R5              ;RESTORE R5
033310  005215                              INC     (R5)


                                    ;*******************************************************************
                                    ;TEST:227       MUL       -1 * #77777 = -1 100001        PS = 10
                                    ;*******************************************************************

033312                              TST227:
033312  012702  177777                      MOV     #-1,%2             ;LOAD MULTIPLICAND WITH -1
033316  070227  077777                      MUL     #77777,%2          ;MULTIPLY -1 * #77777
033322  106737  025414                      MFPS    @#PSWORD           ;SAVE PS
033326  122737  000010  025414              CMPB    #10,@#PSWORD       ;IS PS = 10
033334  001401                              BEQ     11$
```

```
033336  104000                          EMT                             ;PS IS WRONG
033340  022702  177777          11$:    CMP     #-1,%2                  ;IS HIGH ORDER = -1
033344  001401                          BEQ     12$
033346  104000                          EMT                             ;HIGH ORDER IS WRONG
033350  022703  100001          12$:    CMP     #100001,%2!1            ;IS LOW ORDER = 100001
033354  001401                          BEQ     13$
033356  104000                          EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033360                          13$:
033360  005215                          INC     (R5)



                                ;*******************************************************************
                                ;TEST:230       MUL     -2 * #77777 = -1 2          PS = 11
                                ;*******************************************************************

033362                          TST230:
033362  012700  177776                  MOV     #-2,%0                  ;LOAD MULTIPLICAND WITH -2
033366  070027  077777                  MUL     #77777,%0               ;MULTIPLY -2 * #77777
033372  106737  025414                  MFPS    @#PSWORD                ;SAVE PS
033376  122737  000011  025414          CMPB    #11,@#PSWORD            ;IS PS = 11
033404  001401                          BEQ     11$
033406  104000                          EMT                             ;PS IS WRONG
033410  022700  177777          11$:    CMP     #-1,%0                  ;IS HIGH ORDER = -1
033414  001401                          BEQ     12$
033416  104000                          EMT                             ;HIGH ORDER IS WRONG
033420  022701  000002          12$:    CMP     #2,%0!1                 ;IS LOW ORDER = 2
033424  001401                          BEQ     13$
033426  104000                          EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033430                          13$:
033430  005215                          INC     (R5)



                                ;*******************************************************************
                                ;TEST:231       MUL     125252 * #2 = -1 52524          PS = 11
                                ;*******************************************************************

033432                          TST231:
033432  012702  125252                  MOV     #125252,%2              ;LOAD MULTIPLICAND WITH 125252
033436  070227  000002                  MUL     #2,%2                   ;MULTIPLY 125252 * #2
033442  106737  025414                  MFPS    @#PSWORD                ;SAVE PS
033446  122737  000011  025414          CMPB    #11,@#PSWORD            ;IS PS = 11
033454  001401                          BEQ     11$
033456  104000                          EMT                             ;PS IS WRONG
033460  022702  177777          11$:    CMP     #-1,%2                  ;IS HIGH ORDER = -1
033464  001401                          BEQ     12$
033466  104000                          EMT                             ;HIGH ORDER IS WRONG
033470  022703  052524          12$:    CMP     #52524,%2!1             ;IS LOW ORDER = 52524
033474  001401                          BEQ     13$
033476  104000                          EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033500                          13$:
033500  005215                          INC     (R5)



                                ;*******************************************************************
                                ;TEST:232       MUL     125252 * #40000 = 165252 100000         PS = 11
                                ;*******************************************************************
```

```
033502                          TST232:
033502  010501                          MOV     R5,R1           ;SAVE R5
033504  012704  125252                  MOV     #125252,%4      ;LOAD MULTIPLICAND WITH 125252
033510  070427  040000                  MUL     #40000,%4       ;MULTIPLY 125252 * #40000
033514  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
033520  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS PS = 11
033526  001401                          BEQ     11$
033530  104000                          EMT                     ;PS IS WRONG
033532  022704  165252          11$:    CMP     #165252,%4      ;IS HIGH ORDER = 165252
033536  001401                          BEQ     12$
033540  104000                          EMT                     ;HIGH ORDER IS WRONG
033542  022705  100000          12$:    CMP     #100000,%4!1    ;IS LOW ORDER = 100000
033546  001401                          BEQ     13$
033550  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033552                          13$:
033552  010105                          MOV     R1,R5           ;RESTORE R5
033554  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:233       MUL     107070 * #107070 = 31222 26100          PS = 1
                                ;*************************************************************************

033556                          TST233:
033556  012700  107070                  MOV     #107070,%0      ;LOAD MULTIPLICAND WITH 107070
033562  070027  107070                  MUL     #107070,%0      ;MULTIPLY 107070 * #107070
033566  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
033572  122737  000001  025414          CMPB    #1,@#PSWORD     ;IS PS = 1
033600  001401                          BEQ     11$
033602  104000                          EMT                     ;PS IS WRONG
033604  022700  031222          11$:    CMP     #31222,%0       ;IS HIGH ORDER = 31222
033610  001401                          BEQ     12$
033612  104000                          EMT                     ;HIGH ORDER IS WRONG
033614  022701  026100          12$:    CMP     #26100,%0!1     ;IS LOW ORDER = 26100
033620  001401                          BEQ     13$
033622  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033624                          13$:
033624  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:234       MUL     -1 * #1 = -1 -1         PS = 10
                                ;*************************************************************************

033626                          TST234:
033626  012701  177777                  MOV     #-1,%1          ;LOAD MULTIPLICAND WITH -1
033632  070127  000001                  MUL     #1,%1           ;MULTIPLY -1 * #1
033636  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
033642  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS PS = 10
033650  001401                          BEQ     11$
033652  104000                          EMT                     ;PS IS WRONG
033654  022701  177777          11$:    CMP     #-1,%1          ;IS HIGH ORDER = -1
033660  001401                          BEQ     12$
033662  104000                          EMT                     ;HIGH ORDER IS WRONG
033664  022701  177777          12$:    CMP     #-1,%1!1        ;IS LOW ORDER = -1
```

```
033670  001401                               BEQ     13$
033672  104000                               EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033674                               13$:
033674  005215                               INC     (R5)

                                     ;*******************************************************************
                                     ;TEST:235        MUL      -1 * #0 = 0 0         PS = 4
                                     ;*******************************************************************

033676                               TST235:
033676  012703  177777                       MOV     #-1,%3                  ;LOAD MULTIPLICAND WITH -1
033702  070327  000000                       MUL     #0,%3                   ;MULTIPLY -1 * #0
033706  106737  025414                       MFPS    @#PSWORD                ;SAVE PS
033712  122737  000004  025414               CMPB    #4,@#PSWORD             ;IS PS = 4
033720  001401                               BEQ     11$
033722  104000                               EMT                             ;PS IS WRONG
033724  022703  000000               11$:    CMP     #0,%3                   ;IS HIGH ORDER = 0
033730  001401                               BEQ     12$
033732  104000                               EMT                             ;HIGH ORDER IS WRONG
033734  022703  000000               12$:    CMP     #0,%3!1                 ;IS LOW ORDER = 0
033740  001401                               BEQ     13$
033742  104000                               EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
033744                               13$:
033744  005215                               INC     (R5)

                                     ;*******************************************************************
                                     ;TEST:236        MUL      77777 * #100000 = 100000 100000        PS = 11
                                     ;*******************************************************************

033746                               TST236:
033746  010501                               MOV     R5,R1                   ;SAVE R5
033750  012705  077777                       MOV     #77777,%5               ;LOAD MULTIPLICAND WITH 77777
033754  070527  100000                       MUL     #100000,%5              ;MULTIPLY 77777 * #100000
033760  106737  025414                       MFPS    @#PSWORD                ;SAVE PS
033764  122737  000011  025414               CMPB    #11,@#PSWORD            ;IS PS = 11
033772  001401                               BEQ     11$
033774  104000                               EMT                             ;PS IS WRONG
033776  022705  100000               11$:    CMP     #100000,%5              ;IS HIGH ORDER = 100000
034002  001401                               BEQ     12$
034004  104000                               EMT                             ;HIGH ORDER IS WRONG
034006  022705  100000               12$:    CMP     #100000,%5!1            ;IS LOW ORDER = 100000
034012  001401                               BEQ     13$
034014  104000                               EMT                             ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034016                               13$:
034016  010105                               MOV     R1,R5                   ;RESTORE R5
034020  005215                               INC     (R5)

                                     ;*******************************************************************
                                     ;TEST:237        MUL      -1 * #77777 = 100001 100001         PS = 10
                                     ;*******************************************************************

034022                               TST237:
034022  012701  177777                       MOV     #-1,%1                  ;LOAD MULTIPLICAND WITH -1
```

```
034026  070127  077777                   MUL     #77777,%1         ;MULTIPLY -1 * #77777
034032  106737  025414                   MFPS    @#PSWORD          ;SAVE PS
034036  122737  000010  025414           CMPB    #10,@#PSWORD      ;IS PS = 10
034044  001401                           BEQ     11$
034046  104000                           EMT                       ;PS IS WRONG
034050  022701  100001           11$:    CMP     #100001,%1        ;IS HIGH ORDER = 100001
034054  001401                           BEQ     12$
034056  104000                           EMT                       ;HIGH ORDER IS WRONG
034060  022701  100001           12$:    CMP     #100001,%1!1      ;IS LOW ORDER = 100001
034064  001401                           BEQ     13$
034066  104000                           EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034070                           13$:
034070  005215                           INC     (R5)


                                 ;*************************************************************************
                                 ;TEST:240          MUL       77777 * #77777 = 1 1        PS = 1
                                 ;*************************************************************************

034072                           TST240:
034072  012703  077777                   MOV     #77777,%3         ;LOAD MULTIPLICAND WITH 77777
034076  070327  077777                   MUL     #77777,%3         ;MULTIPLY 77777 * #77777
034102  106737  025414                   MFPS    @#PSWORD          ;SAVE PS
034106  122737  000001  025414           CMPB    #1,@#PSWORD       ;IS PS = 1
034114  001401                           BEQ     11$
034116  104000                           EMT                       ;PS IS WRONG
034120  022703  000001           11$:    CMP     #1,%3             ;IS HIGH ORDER = 1
034124  001401                           BEQ     12$
034126  104000                           EMT                       ;HIGH ORDER IS WRONG
034130  022703  000001           12$:    CMP     #1,%3!1           ;IS LOW ORDER = 1
034134  001401                           BEQ     13$
034136  104000                           EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034140                           13$:
034140  005215                           INC     (R5)


                                 ;*************************************************************************
                                 ;TEST:241          MUL       2 * #2 = 4 4        PS = 0
                                 ;*************************************************************************

034142                           TST241:
034142  010501                           MOV     R5,R1             ;SAVE R5
034144  012705  000002                   MOV     #2,%5             ;LOAD MULTIPLICAND WITH 2
034150  070527  000002                   MUL     #2,%5             ;MULTIPLY 2 * #2
034154  106737  025414                   MFPS    @#PSWORD          ;SAVE PS
034160  122737  000000  025414           CMPB    #0,@#PSWORD       ;IS PS = 0
034166  001401                           BEQ     11$
034170  104000                           EMT                       ;PS IS WRONG
034172  022705  000004           11$:    CMP     #4,%5             ;IS HIGH ORDER = 4
034176  001401                           BEQ     12$
034200  104000                           EMT                       ;HIGH ORDER IS WRONG
034202  022705  000004           12$:    CMP     #4,%5!1           ;IS LOW ORDER = 4
034206  001401                           BEQ     13$
034210  104000                           EMT                       ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034212                           13$:
034212  010105                           MOV     R1,R5             ;RESTORE R5
```

```
034214  005215                          INC     (R5)


034216  012702  040000                  MOV     #40000,%2
034222  012703  025442                  MOV     #S5,%3
034226  012704  025444                  MOV     #S6,%4

                                ;*************************************************************************
                                ;TEST:242       MUL     125252 * S5 = 165252 100000         PS = 11
                                ;*************************************************************************

034232                          TST242:
034232  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034236  070067  171200                  MUL     S5,%0           ;MULTIPLY 125252 * S5
034242  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
034246  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS PS = 11
034254  001401                          BEQ     11$
034256  104000                          EMT                     ;PS IS WRONG
034260  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034264  001401                          BEQ     12$
034266  104000                          EMT                     ;HIGH ORDER IS WRONG
034270  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034274  001401                          BEQ     13$
034276  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034300                          13$:
034300  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:243       MUL     125252 * @S6 = 165252 100000        PS = 11
                                ;*************************************************************************

034302                          TST243:
034302  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034306  070077  171132                  MUL     @S6,%0          ;MULTIPLY 125252 * @S6
034312  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
034316  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS PS = 11
034324  001401                          BEQ     11$
034326  104000                          EMT                     ;PS IS WRONG
034330  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034334  001401                          BEQ     12$
034336  104000                          EMT                     ;HIGH ORDER IS WRONG
034340  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034344  001401                          BEQ     13$
034346  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034350                          13$:
034350  005215                          INC     (R5)


                                ;*************************************************************************
                                ;TEST:244       MUL     125252 * @#S5 = 165252 100000       PS = 11
                                ;*************************************************************************

034352                          TST244:
034352  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034356  070037  025442                  MUL     @#S5,%0         ;MULTIPLY 125252 * @#S5
```

```
034362 106737 025414                 MFPS   @#PSWORD         ;SAVE PS
034366 122737 000011 025414          CMPB   #11,@#PSWORD     ;IS PS = 11
034374 001401                        BEQ    11$
034376 104000                        EMT                     ;PS IS WRONG
034400 022700 165252          11$:   CMP    #165252,%0       ;IS HIGH ORDER = 165252
034404 001401                        BEQ    12$
034406 104000                        EMT                     ;HIGH ORDER IS WRONG
034410 022701 100000          12$:   CMP    #100000,%0!1     ;IS LOW ORDER = 100000
034414 001401                        BEQ    13$
034416 104000                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034420                        13$:
034420 005215                        INC    (R5)


                              ;*******************************************************************
                              ;TEST:245      MUL      125252 * %2 = 165252 100000        PS = 11
                              ;*******************************************************************

034422                        TST245:
034422 012700 125252                 MOV    #125252,%0       ;LOAD MULTIPLICAND WITH 125252
034426 070002                        MUL    %2,%0            ;MULTIPLY 125252 * %2
034430 106737 025414                 MFPS   @#PSWORD         ;SAVE PS
034434 122737 000011 025414          CMPB   #11,@#PSWORD     ;IS PS = 11
034442 001401                        BEQ    11$
034444 104000                        EMT                     ;PS IS WRONG
034446 022700 165252          11$:   CMP    #165252,%0       ;IS HIGH ORDER = 165252
034452 001401                        BEQ    12$
034454 104000                        EMT                     ;HIGH ORDER IS WRONG
034456 022701 100000          12$:   CMP    #100000,%0!1     ;IS LOW ORDER = 100000
034462 001401                        BEQ    13$
034464 104000                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034466                        13$:
034466 005215                        INC    (R5)


                              ;*******************************************************************
                              ;TEST:246      MUL      125252 * (3)+ = 165252 100000        PS = 11
                              ;*******************************************************************

034470                        TST246:
034470 012700 125252                 MOV    #125252,%0       ;LOAD MULTIPLICAND WITH 125252
034474 070023                        MUL    (3)+,%0          ;MULTIPLY 125252 * (3)+
034476 106737 025414                 MFPS   @#PSWORD         ;SAVE PS
034502 122737 000011 025414          CMPB   #11,@#PSWORD     ;IS PS = 11
034510 001401                        BEQ    11$
034512 104000                        EMT                     ;PS IS WRONG
034514 022700 165252          11$:   CMP    #165252,%0       ;IS HIGH ORDER = 165252
034520 001401                        BEQ    12$
034522 104000                        EMT                     ;HIGH ORDER IS WRONG
034524 022701 100000          12$:   CMP    #100000,%0!1     ;IS LOW ORDER = 100000
034530 001401                        BEQ    13$
034532 104000                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034534                        13$:
034534 005215                        INC    (R5)
```

```
                                   ;*******************************************************************
                                   ;TEST:247       MUL     125252 * -(3) = 165252 100000        PS = 11
                                   ;*******************************************************************

034536                             TST247:
034536  012700  125252                     MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034542  070043                             MUL     -(3),%0         ;MULTIPLY 125252 * -(3)
034544  106737  025414                     MFPS    @#PSWORD        ;SAVE PS
034550  122737  000011  025414             CMPB    #11,@#PSWORD    ;IS PS = 11
034556  001401                             BEQ     11$
034560  104000                             EMT                     ;PS IS WRONG
034562  022700  165252             11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034566  001401                             BEQ     12$
034570  104000                             EMT                     ;HIGH ORDER IS WRONG
034572  022701  100000             12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034576  001401                             BEQ     13$
034600  104000                             EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034602                             13$:
034602  005215                             INC     (R5)


                                   ;*******************************************************************
                                   ;TEST:250       MUL     125252 * 2(4) = 165252 100000        PS = 11
                                   ;*******************************************************************

034604                             TST250:
034604  012700  125252                     MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034610  070064  000002                     MUL     2(4),%0         ;MULTIPLY 125252 * 2(4)
034614  106737  025414                     MFPS    @#PSWORD        ;SAVE PS
034620  122737  000011  025414             CMPB    #11,@#PSWORD    ;IS PS = 11
034626  001401                             BEQ     11$
034630  104000                             EMT                     ;PS IS WRONG
034632  022700  165252             11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034636  001401                             BEQ     12$
034640  104000                             EMT                     ;HIGH ORDER IS WRONG
034642  022701  100000             12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034646  001401                             BEQ     13$
034650  104000                             EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034652                             13$:
034652  005215                             INC     (R5)


                                   ;*******************************************************************
                                   ;TEST:251       MUL     125252 * @(4) = 165252 100000        PS = 11
                                   ;*******************************************************************

034654                             TST251:
034654  012700  125252                     MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034660  070074  000000                     MUL     @(4),%0         ;MULTIPLY 125252 * @(4)
034664  106737  025414                     MFPS    @#PSWORD        ;SAVE PS
034670  122737  000011  025414             CMPB    #11,@#PSWORD    ;IS PS = 11
034676  001401                             BEQ     11$
034700  104000                             EMT                     ;PS IS WRONG
034702  022700  165252             11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034706  001401                             BEQ     12$
034710  104000                             EMT                     ;HIGH ORDER IS WRONG
```

```
034712  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034716  001401                          BEQ     13$
034720  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034722                          13$:
034722  005215                          INC     (R5)



                                ;*******************************************************************
                                ;TEST:252       MUL     125252 * @(4)+ = 165252 100000        PS = 11
                                ;*******************************************************************

034724                          TST252:
034724  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034730  070034                          MUL     @(4)+,%0        ;MULTIPLY 125252 * @(4)+
034732  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
034736  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS PS = 11
034744  001401                          BEQ     11$
034746  104000                          EMT                     ;PS IS WRONG
034750  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
034754  001401                          BEQ     12$
034756  104000                          EMT                     ;HIGH ORDER IS WRONG
034760  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
034764  001401                          BEQ     13$
034766  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
034770                          13$:
034770  005215                          INC     (R5)



                                ;*******************************************************************
                                ;TEST:253       MUL     125252 * @-(4) = 165252 100000        PS = 11
                                ;*******************************************************************

034772                          TST253:
034772  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
034776  070054                          MUL     @-(4),%0        ;MULTIPLY 125252 * @-(4)
035000  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
035004  122737  000011  025414          CMPB    #11,@#PSWORD    ;IS PS = 11
035012  001401                          BEQ     11$
035014  104000                          EMT                     ;PS IS WRONG
035016  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
035022  001401                          BEQ     12$
035024  104000                          EMT                     ;HIGH ORDER IS WRONG
035026  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
035032  001401                          BEQ     13$
035034  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
035036                          13$:
035036  005215                          INC     (R5)
```

```
                                               ;*********************************
                                               ;       DIV INSTRUCTION TESTS
                                               ;*********************************



                                ;*********************************************************************
                                ;TEST:254        DIV     0 4 / #2 = 2          REM = 0       PS = 0
                                ;*********************************************************************

035040                          TST254:
035040  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
035044  012701  000004                  MOV     #4,%0+1         ;LOAD LOW ORDER WITH 4
035050  071027  000002                  DIV     #2,%0           ;DIVIDE BY #2
035054  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
035060  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
035066  001401                          BEQ     11$
035070  104000                          EMT                     ;PS IS WRONG
035072  022700  000002          11$:    CMP     #2,%0           ;IS QUOTIENT = 2
035076  001401                          BEQ     12$
035100  104000                          EMT                     ;QUOTIENT IS WRONG
035102  022701  000000          12$:    CMP     #0,%0+1         ;IS REMAINDER = 0
035106  001401                          BEQ     13$
035110  104000                          EMT                     ;WRONG REMAINDER
035112                          13$:
035112  005215                          INC     (R5)

                                ;*********************************************************************
                                ;TEST:255        DIV     -1 -9. / #3 = -3          REM = 0       PS = 10
                                ;*********************************************************************

035114                          TST255:
035114  012702  177777                  MOV     #-1,%2          ;LOAD HIGH ORDER WITH -1
035120  012703  177767                  MOV     #-9.,%2+1       ;LOAD LOW ORDER WITH -9.
035124  071227  000003                  DIV     #3,%2           ;DIVIDE BY #3
035130  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
035134  122737  000010  025414          CMPB    #10,@#PSWORD    ;IS PS = 10
035142  001401                          BEQ     11$
035144  104000                          EMT                     ;PS IS WRONG
035146  022702  177775          11$:    CMP     #-3,%2          ;IS QUOTIENT = -3
035152  001401                          BEQ     12$
035154  104000                          EMT                     ;QUOTIENT IS WRONG
035156  022703  000000          12$:    CMP     #0,%2+1         ;IS REMAINDER = 0
035162  001401                          BEQ     13$
035164  104000                          EMT                     ;WRONG REMAINDER
035166                          13$:
035166  005215                          INC     (R5)

                                ;*********************************************************************
                                ;TEST:256        DIV     0 9. / #2 = 4          REM = 1       PS = 0
                                ;*********************************************************************

035170                          TST256:
035170  010501                          MOV     R5,R1           ;SAVE R5
035172  012704  000000                  MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
035176  012705  000011                  MOV     #9.,%4+1        ;LOAD LOW ORDER WITH 9.
```

```
035202 071427 000002                    DIV     #2,%4                   ;DIVIDE BY #2
035206 106737 025414                    MFPS    @#PSWORD                ;SAVE PS
035212 122737 000000 025414             CMPB    #0,@#PSWORD             ;IS PS = 0
035220 001401                           BEQ     11$
035222 104000                           EMT                             ;PS IS WRONG
035224 022704 000004            11$:    CMP     #4,%4                   ;IS QUOTIENT = 4
035230 001401                           BEQ     12$
035232 104000                           EMT                             ;QUOTIENT IS WRONG
035234 022705 000001            12$:    CMP     #1,%4+1                 ;IS REMAINDER = 1
035240 001401                           BEQ     13$
035242 104000                           EMT                             ;WRONG REMAINDER
035244                          13$:
035244 010105                           MOV     R1,R5                   ;RESTORE R5
035246 005215                           INC     (R5)

                                ;*******************************************************************
                                ;TEST:257        DIV     -1 -9. / #2 = -4        REM = -1        PS = 10
                                ;*******************************************************************

035250                          TST257:
035250 012700 177777                    MOV     #-1,%0                  ;LOAD HIGH ORDER WITH -1
035254 012701 177767                    MOV     #-9.,%0+1               ;LOAD LOW ORDER WITH -9.
035260 071027 000002                    DIV     #2,%0                   ;DIVIDE BY #2
035264 106737 025414                    MFPS    @#PSWORD                ;SAVE PS
035270 122737 000010 025414             CMPB    #10,@#PSWORD            ;IS PS = 10
035276 001401                           BEQ     11$
035300 104000                           EMT                             ;PS IS WRONG
035302 022700 177774            11$:    CMP     #-4,%0                  ;IS QUOTIENT = -4
035306 001401                           BEQ     12$
035310 104000                           EMT                             ;QUOTIENT IS WRONG
035312 022701 177777            12$:    CMP     #-1,%0+1                ;IS REMAINDER = -1
035316 001401                           BEQ     13$
035320 104000                           EMT                             ;WRONG REMAINDER
035322                          13$:
035322 005215                           INC     (R5)

                                ;*******************************************************************
                                ;TEST:260        DIV     0 2 / #-3 = 0        REM = 2        PS = 4
                                ;*******************************************************************

035324                          TST260:
035324 012702 000000                    MOV     #0,%2                   ;LOAD HIGH ORDER WITH 0
035330 012703 000002                    MOV     #2,%2+1                 ;LOAD LOW ORDER WITH 2
035334 071227 177775                    DIV     #-3,%2                  ;DIVIDE BY #-3
035340 106737 025414                    MFPS    @#PSWORD                ;SAVE PS
035344 122737 000004 025414             CMPB    #4,@#PSWORD             ;IS PS = 4
035352 001401                           BEQ     11$
035354 104000                           EMT                             ;PS IS WRONG
035356 022702 000000            11$:    CMP     #0,%2                   ;IS QUOTIENT = 0
035362 001401                           BEQ     12$
035364 104000                           EMT                             ;QUOTIENT IS WRONG
035366 022703 000002            12$:    CMP     #2,%2+1                 ;IS REMAINDER = 2
035372 001401                           BEQ     13$
035374 104000                           EMT                             ;WRONG REMAINDER
035376                          13$:
035376 005215                           INC     (R5)
```

```
                                       ;*********************************************************************
                                       ;TEST:261      DIV     -1 -2 / #3 = 0        REM = -2       PS = 4
                                       ;*********************************************************************

035400                                 TST261:
035400  010501                                 MOV     R5,R1                   ;SAVE R5
035402  012704  177777                         MOV     #-1,%4                  ;LOAD HIGH ORDER WITH -1
035406  012705  177776                         MOV     #-2,%4+1                ;LOAD LOW ORDER WITH -2
035412  071427  000003                         DIV     #3,%4                   ;DIVIDE BY #3
035416  106737  025414                         MFPS    @#PSWORD                ;SAVE PS
035422  122737  000004  025414                 CMPB    #4,@#PSWORD             ;IS PS = 4
035430  001401                                 BEQ     11$
035432  104000                                 EMT                             ;PS IS WRONG
035434  022704  000000                 11$:    CMP     #0,%4                   ;IS QUOTIENT = 0
035440  001401                                 BEQ     12$
035442  104000                                 EMT                             ;QUOTIENT IS WRONG
035444  022705  177776                 12$:    CMP     #-2,%4+1                ;IS REMAINDER = -2
035450  001401                                 BEQ     13$
035452  104000                                 EMT                             ;WRONG REMAINDER
035454                                 13$:
035454  010105                                 MOV     R1,R5                   ;RESTORE R5
035456  005215                                 INC     (R5)

                                       ;*********************************************************************
                                       ;TEST:262      DIV     -1 -1 / #1 = -1        REM = 0       PS = 10
                                       ;*********************************************************************

035460                                 TST262:
035460  012700  177777                         MOV     #-1,%0                  ;LOAD HIGH ORDER WITH -1
035464  012701  177777                         MOV     #-1,%0+1                ;LOAD LOW ORDER WITH -1
035470  071027  000001                         DIV     #1,%0                   ;DIVIDE BY #1
035474  106737  025414                         MFPS    @#PSWORD                ;SAVE PS
035500  122737  000010  025414                 CMPB    #10,@#PSWORD            ;IS PS = 10
035506  001401                                 BEQ     11$
035510  104000                                 EMT                             ;PS IS WRONG
035512  022700  177777                 11$:    CMP     #-1,%0                  ;IS QUOTIENT = -1
035516  001401                                 BEQ     12$
035520  104000                                 EMT                             ;QUOTIENT IS WRONG
035522  022701  000000                 12$:    CMP     #0,%0+1                 ;IS REMAINDER = 0
035526  001401                                 BEQ     13$
035530  104000                                 EMT                             ;WRONG REMAINDER
035532                                 13$:
035532  005215                                 INC     (R5)

                                       ;*********************************************************************
                                       ;TEST:263      DIV     0 0 / #1 = 0        REM = 0       PS = 4
                                       ;*********************************************************************

035534                                 TST263:
035534  012700  000000                         MOV     #0,%0                   ;LOAD HIGH ORDER WITH 0
035540  012701  000000                         MOV     #0,%0+1                 ;LOAD LOW ORDER WITH 0
035544  071027  000001                         DIV     #1,%0                   ;DIVIDE BY #1
035550  106737  025414                         MFPS    @#PSWORD                ;SAVE PS
035554  122737  000004  025414                 CMPB    #4,@#PSWORD             ;IS PS = 4
035562  001401                                 BEQ     11$
```

```
035564  104000                          EMT                             ;PS IS WRONG
035566  022700  000000          11$:    CMP     #0,%0                   ;IS QUOTIENT = 0
035572  001401                          BEQ     12$
035574  104000                          EMT                             ;QUOTIENT IS WRONG
035576  022701  000000          12$:    CMP     #0,%0+1                 ;IS REMAINDER = 0
035602  001401                          BEQ     13$
035604  104000                          EMT                             ;WRONG REMAINDER
035606                          13$:
035606  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:264       DIV     -1 125252 / #2 = 152525         REM = 0       PS = 10
                                ;*************************************************************************

035610                          TST264:
035610  012702  177777                  MOV     #-1,%2                  ;LOAD HIGH ORDER WITH -1
035614  012703  125252                  MOV     #125252,%2+1            ;LOAD LOW ORDER WITH 125252
035620  071227  000002                  DIV     #2,%2                   ;DIVIDE BY #2
035624  106737  025414                  MFPS    @#PSWORD                ;SAVE PS
035630  122737  000010  025414          CMPB    #10,@#PSWORD            ;IS PS = 10
035636  001401                          BEQ     11$
035640  104000                          EMT                             ;PS IS WRONG
035642  022702  152525          11$:    CMP     #152525,%2              ;IS QUOTIENT = 152525
035646  001401                          BEQ     12$
035650  104000                          EMT                             ;QUOTIENT IS WRONG
035652  022703  000000          12$:    CMP     #0,%2+1                 ;IS REMAINDER = 0
035656  001401                          BEQ     13$
035660  104000                          EMT                             ;WRONG REMAINDER
035662                          13$:
035662  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:265       DIV     -1 -1 / #-1 = 1         REM = 0       PS = 0
                                ;*************************************************************************

035664                          TST265:
035664  010501                          MOV     R5,R1                   ;SAVE R5
035666  012704  177777                  MOV     #-1,%4                  ;LOAD HIGH ORDER WITH -1
035672  012705  177777                  MOV     #-1,%4+1                ;LOAD LOW ORDER WITH -1
035676  071427  177777                  DIV     #-1,%4                  ;DIVIDE BY #-1
035702  106737  025414                  MFPS    @#PSWORD                ;SAVE PS
035706  122737  000000  025414          CMPB    #0,@#PSWORD             ;IS PS = 0
035714  001401                          BEQ     11$
035716  104000                          EMT                             ;PS IS WRONG
035720  022704  000001          11$:    CMP     #1,%4                   ;IS QUOTIENT = 1
035724  001401                          BEQ     12$
035726  104000                          EMT                             ;QUOTIENT IS WRONG
035730  022705  000000          12$:    CMP     #0,%4+1                 ;IS REMAINDER = 0
035734  001401                          BEQ     13$
035736  104000                          EMT                             ;WRONG REMAINDER
035740                          13$:
035740  010105                          MOV     R1,R5                   ;RESTORE R5
035742  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:266       DIV     25253 1 / #125252 = 100000      REM = 1       PS = 10
```

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

        035744                      TST266:
        035744  012700  025253              MOV     #25253,%0       ;LOAD HIGH ORDER WITH 25253
        035750  012701  000001              MOV     #1,%0+1         ;LOAD LOW ORDER WITH 1
        035754  071027  125252              DIV     #125252,%0      ;DIVIDE BY #125252
        035760  106737  025414              MFPS    @#PSWORD        ;SAVE PS
        035764  122737  000010  025414      CMPB    #10,@#PSWORD    ;IS PS = 10
        035772  001401                      BEQ     11$
        035774  104000                      EMT                     ;PS IS WRONG
        035776  022700  100000      11$:    CMP     #100000,%0      ;IS QUOTIENT = 100000
        036002  001401                      BEQ     12$
        036004  104000                      EMT                     ;QUOTIENT IS WRONG
        036006  022701  000001      12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
        036012  001401                      BEQ     13$
        036014  104000                      EMT                     ;WRONG REMAINDER
        036016                      13$:
        036016  005215                      INC     (R5)

                                    ;*******************************************************************************
                                    ;TEST:267       DIV     37777 77777 / #77777 = 77777        REM = 77776         PS = 0
                                    ;*******************************************************************************

        036020                      TST267:
        036020  012702  037777              MOV     #37777,%2       ;LOAD HIGH ORDER WITH 37777
        036024  012703  077777              MOV     #77777,%2+1     ;LOAD LOW ORDER WITH 77777
        036030  071227  077777              DIV     #77777,%2       ;DIVIDE BY #77777
        036034  106737  025414              MFPS    @#PSWORD        ;SAVE PS
        036040  122737  000000  025414      CMPB    #0,@#PSWORD     ;IS PS = 0
        036046  001401                      BEQ     11$
        036050  104000                      EMT                     ;PS IS WRONG
        036052  022702  077777      11$:    CMP     #77777,%2       ;IS QUOTIENT = 77777
        036056  001401                      BEQ     12$
        036060  104000                      EMT                     ;QUOTIENT IS WRONG
        036062  022703  077776      12$:    CMP     #77776,%2+1     ;IS REMAINDER = 77776
        036066  001401                      BEQ     13$
        036070  104000                      EMT                     ;WRONG REMAINDER
        036072                      13$:
        036072  005215                      INC     (R5)

                                    ;*******************************************************************************
                                    ;TEST:270       DIV     0 100000 / #2 = 40000        REM = 0      PS = 0
                                    ;*******************************************************************************

        036074                      TST270:
        036074  010501                      MOV     R5,R1           ;SAVE R5
        036076  012704  000000              MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
        036102  012705  100000              MOV     #100000,%4+1    ;LOAD LOW ORDER WITH 100000
        036106  071427  000002              DIV     #2,%4           ;DIVIDE BY #2
        036112  106737  025414              MFPS    @#PSWORD        ;SAVE PS
        036116  122737  000000  025414      CMPB    #0,@#PSWORD     ;IS PS = 0
        036124  001401                      BEQ     11$
        036126  104000                      EMT                     ;PS IS WRONG
        036130  022704  040000      11$:    CMP     #40000,%4       ;IS QUOTIENT = 40000
        036134  001401                      BEQ     12$
        036136  104000                      EMT                     ;QUOTIENT IS WRONG
```

```
036140  022705  000000          12$:    CMP     #0,%4+1         ;IS REMAINDER = 0
036144  001401                          BEQ     13$
036146  104000                          EMT                     ;WRONG REMAINDER
036150                          13$:
036150  010105                          MOV     R1,R5           ;RESTORE R5
036152  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:271       DIV     177777 77777 / #177776 = 40000       REM = 177777      PS = 0
                                ;*************************************************************************

036154                          TST271:
036154  012700  177777                  MOV     #177777,%0      ;LOAD HIGH ORDER WITH 177777
036160  012701  077777                  MOV     #77777,%0+1     ;LOAD LOW ORDER WITH 77777
036164  071027  177776                  DIV     #177776,%0      ;DIVIDE BY #177776
036170  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
036174  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
036202  001401                          BEQ     11$
036204  104000                          EMT                     ;PS IS WRONG
036206  022700  040000          11$:    CMP     #40000,%0       ;IS QUOTIENT = 40000
036212  001401                          BEQ     12$
036214  104000                          EMT                     ;QUOTIENT IS WRONG
036216  022701  177777          12$:    CMP     #177777,%0+1    ;IS REMAINDER = 177777
036222  001401                          BEQ     13$
036224  104000                          EMT                     ;WRONG REMAINDER
036226                          13$:
036226  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:272       DIV     0 52525 / #52525 = 1        REM = 0      PS = 0
                                ;*************************************************************************

036230                          TST272:
036230  012702  000000                  MOV     #0,%2           ;LOAD HIGH ORDER WITH 0
036234  012703  052525                  MOV     #52525,%2+1     ;LOAD LOW ORDER WITH 52525
036240  071227  052525                  DIV     #52525,%2       ;DIVIDE BY #52525
036244  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
036250  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
036256  001401                          BEQ     11$
036260  104000                          EMT                     ;PS IS WRONG
036262  022702  000001          11$:    CMP     #1,%2           ;IS QUOTIENT = 1
036266  001401                          BEQ     12$
036270  104000                          EMT                     ;QUOTIENT IS WRONG
036272  022703  000000          12$:    CMP     #0,%2+1         ;IS REMAINDER = 0
036276  001401                          BEQ     13$
036300  104000                          EMT                     ;WRONG REMAINDER
036302                          13$:
036302  005215                          INC     (R5)

                                ;*************************************************************************
                                ;TEST:273       DIV     0 77777 / #0 = DUMMY        REM = DUMMY      PS = 3
                                ;*************************************************************************

036304                          TST273:
036304  010501                          MOV     R5,R1           ;SAVE R5
036306  012704  000000                  MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
```

```
036312  012705  077777                   MOV     #77777,%4+1      ;LOAD LOW ORDER WITH 77777
036316  071427  000000                   DIV     #0,%4            ;DIVIDE BY #0
036322  106737  025414                   MFPS    @#PSWORD         ;SAVE PS
036326  042737  000014  025414           BIC     #14,@#PSWORD
036334  122737  000003  025414           CMPB    #3,@#PSWORD      ;IS PS = 3
036342  001401                           BEQ     13$
036344  104000                           EMT                      ;PS IS WRONG
036346                           13$:
036346  010105                           MOV     R1,R5            ;RESTORE R5
036350  005215                           INC     (R5)

                                 ;*******************************************************************
                                 ;TEST:274      DIV     77777 177777 / #2 = DUMMY       REM = DUMMY      PS = 2
                                 ;*******************************************************************

036352                           TST274:
036352  012700  077777                   MOV     #77777,%0        ;LOAD HIGH ORDER WITH 77777
036356  012701  177777                   MOV     #177777,%0+1     ;LOAD LOW ORDER WITH 177777
036362  071027  000002                   DIV     #2,%0            ;DIVIDE BY #2
036366  106737  025414                   MFPS    @#PSWORD         ;SAVE PS
036372  042737  000014  025414           BIC     #14,@#PSWORD
036400  122737  000002  025414           CMPB    #2,@#PSWORD      ;IS PS = 2
036406  001401                           BEQ     13$
036410  104000                           EMT                      ;PS IS WRONG
036412                           13$:
036412  005215                           INC     (R5)

036414  012702  000002                   MOV     #2,%2
036420  012703  025452                   MOV     #S9,%3
036424  012704  025454                   MOV     #S10,%4

                                 ;*******************************************************************
                                 ;TEST:275      DIV     0 52525 / S9 = 25252       REM = 1      PS = 0
                                 ;*******************************************************************

036430                           TST275:
036430  012700  000000                   MOV     #0,%0            ;LOAD HIGH ORDER WITH 0
036434  012701  052525                   MOV     #52525,%0+1      ;LOAD LOW ORDER WITH 52525
036440  071067  167006                   DIV     S9,%0            ;DIVIDE BY S9
036444  106737  025414                   MFPS    @#PSWORD         ;SAVE PS
036450  122737  000000  025414           CMPB    #0,@#PSWORD      ;IS PS = 0
036456  001401                           BEQ     11$
036460  104000                           EMT                      ;PS IS WRONG
036462  022700  025252           11$:    CMP     #25252,%0        ;IS QUOTIENT = 25252
036466  001401                           BEQ     12$
036470  104000                           EMT                      ;QUOTIENT IS WRONG
036472  022701  000001           12$:    CMP     #1,%0+1          ;IS REMAINDER = 1
036476  001401                           BEQ     13$
036500  104000                           EMT                      ;WRONG REMAINDER
036502                           13$:
036502  005215                           INC     (R5)

                                 ;*******************************************************************
                                 ;TEST:276      DIV     0 52525 / @S10 = 25252       REM = 1      PS = 0
                                 ;*******************************************************************
```

```
036504                          TST276:
036504  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
036510  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
036514  071077  166734                  DIV     @S10,%0         ;DIVIDE BY @S10
036520  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
036524  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
036532  001401                          BEQ     11$
036534  104000                          EMT                     ;PS IS WRONG
036536  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
036542  001401                          BEQ     12$
036544  104000                          EMT                     ;QUOTIENT IS WRONG
036546  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
036552  001401                          BEQ     13$
036554  104000                          EMT                     ;WRONG REMAINDER
036556                          13$:
036556  005215                          INC     (R5)

                                ;*******************************************************************
                                ;TEST:277       DIV     0 52525 / @#S9 = 25252      REM = 1     PS = 0
                                ;*******************************************************************

036560                          TST277:
036560  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
036564  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
036570  071037  025452                  DIV     @#S9,%0         ;DIVIDE BY @#S9
036574  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
036600  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
036606  001401                          BEQ     11$
036610  104000                          EMT                     ;PS IS WRONG
036612  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
036616  001401                          BEQ     12$
036620  104000                          EMT                     ;QUOTIENT IS WRONG
036622  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
036626  001401                          BEQ     13$
036630  104000                          EMT                     ;WRONG REMAINDER
036632                          13$:
036632  005215                          INC     (R5)

                                ;*******************************************************************
                                ;TEST:300       DIV     0 52525 / %2 = 25252      REM = 1     PS = 0
                                ;*******************************************************************

036634                          TST300:
036634  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
036640  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
036644  071002                          DIV     %2,%0           ;DIVIDE BY %2
036646  106737  025414                  MFPS    @#PSWORD        ;SAVE PS
036652  122737  000000  025414          CMPB    #0,@#PSWORD     ;IS PS = 0
036660  001401                          BEQ     11$
036662  104000                          EMT                     ;PS IS WRONG
036664  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
036670  001401                          BEQ     12$
036672  104000                          EMT                     ;QUOTIENT IS WRONG
036674  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
036700  001401                          BEQ     13$
036702  104000                          EMT                     ;WRONG REMAINDER
```

```
036704                          13$:
036704  005215                          INC     (R5)

                                ;*******************************************************************
                                ;TEST:301       DIV     0 52525 / (3)+ = 25252       REM = 1     PS = 0
                                ;*******************************************************************

036706                          TST301:
036706  012700  000000                  MOV     #0,%0            ;LOAD HIGH ORDER WITH 0
036712  012701  052525                  MOV     #52525,%0+1      ;LOAD LOW ORDER WITH 52525
036716  071023                          DIV     (3)+,%0          ;DIVIDE BY (3)+
036720  106737  025414                  MFPS    @#PSWORD         ;SAVE PS
036724  122737  000000  025414          CMPB    #0,@#PSWORD      ;IS PS = 0
036732  001401                          BEQ     11$
036734  104000                          EMT                      ;PS IS WRONG
036736  022700  025252          11$:    CMP     #25252,%0        ;IS QUOTIENT = 25252
036742  001401                          BEQ     12$
036744  104000                          EMT                      ;QUOTIENT IS WRONG
036746  022701  000001          12$:    CMP     #1,%0+1          ;IS REMAINDER = 1
036752  001401                          BEQ     13$
036754  104000                          EMT                      ;WRONG REMAINDER
036756                          13$:
036756  005215                          INC     (R5)

                                ;*******************************************************************
                                ;TEST:302       DIV     0 52525 / -(3) = 25252       REM = 1     PS = 0
                                ;*******************************************************************

036760                          TST302:
036760  012700  000000                  MOV     #0,%0            ;LOAD HIGH ORDER WITH 0
036764  012701  052525                  MOV     #52525,%0+1      ;LOAD LOW ORDER WITH 52525
036770  071043                          DIV     -(3),%0          ;DIVIDE BY -(3)
036772  106737  025414                  MFPS    @#PSWORD         ;SAVE PS
036776  122737  000000  025414          CMPB    #0,@#PSWORD      ;IS PS = 0
037004  001401                          BEQ     11$
037006  104000                          EMT                      ;PS IS WRONG
037010  022700  025252          11$:    CMP     #25252,%0        ;IS QUOTIENT = 25252
037014  001401                          BEQ     12$
037016  104000                          EMT                      ;QUOTIENT IS WRONG
037020  022701  000001          12$:    CMP     #1,%0+1          ;IS REMAINDER = 1
037024  001401                          BEQ     13$
037026  104000                          EMT                      ;WRONG REMAINDER
037030                          13$:
037030  005215                          INC     (R5)

                                ;*******************************************************************
                                ;TEST:303       DIV     0 52525 / 2(4) = 25252       REM = 1     PS = 0
                                ;*******************************************************************

037032                          TST303:
037032  012700  000000                  MOV     #0,%0            ;LOAD HIGH ORDER WITH 0
037036  012701  052525                  MOV     #52525,%0+1      ;LOAD LOW ORDER WITH 52525
037042  071064  000002                  DIV     2(4),%0          ;DIVIDE BY 2(4)
037046  106737  025414                  MFPS    @#PSWORD         ;SAVE PS
037052  122737  000000  025414          CMPB    #0,@#PSWORD      ;IS PS = 0
037060  001401                          BEQ     11$
```

```
037062  104000                            EMT                            ;PS IS WRONG
037064  022700  025252            11$:    CMP     #25252,%0              ;IS QUOTIENT = 25252
037070  001401                            BEQ     12$
037072  104000                            EMT                            ;QUOTIENT IS WRONG
037074  022701  000001            12$:    CMP     #1,%0+1                ;IS REMAINDER = 1
037100  001401                            BEQ     13$
037102  104000                            EMT                            ;WRONG REMAINDER
037104                            13$:
037104  005215                            INC     (R5)

                                  ;*******************************************************************
                                  ;TEST:304      DIV     0 52525 / @(4) = 25252      REM = 1     PS = 0
                                  ;*******************************************************************

037106                            TST304:
037106  012700  000000                    MOV     #0,%0                  ;LOAD HIGH ORDER WITH 0
037112  012701  052525                    MOV     #52525,%0+1            ;LOAD LOW ORDER WITH 52525
037116  071074  000000                    DIV     @(4),%0                ;DIVIDE BY @(4)
037122  106737  025414                    MFPS    @#PSWORD               ;SAVE PS
037126  122737  000000  025414            CMPB    #0,@#PSWORD            ;IS PS = 0
037134  001401                            BEQ     11$
037136  104000                            EMT                            ;PS IS WRONG
037140  022700  025252            11$:    CMP     #25252,%0              ;IS QUOTIENT = 25252
037144  001401                            BEQ     12$
037146  104000                            EMT                            ;QUOTIENT IS WRONG
037150  022701  000001            12$:    CMP     #1,%0+1                ;IS REMAINDER = 1
037154  001401                            BEQ     13$
037156  104000                            EMT                            ;WRONG REMAINDER
037160                            13$:
037160  005215                            INC     (R5)

                                  ;*******************************************************************
                                  ;TEST:305      DIV     0 52525 / @(4)+ = 25252      REM = 1     PS = 0
                                  ;*******************************************************************

037162                            TST305:
037162  012700  000000                    MOV     #0,%0                  ;LOAD HIGH ORDER WITH 0
037166  012701  052525                    MOV     #52525,%0+1            ;LOAD LOW ORDER WITH 52525
037172  071034                            DIV     @(4)+,%0               ;DIVIDE BY @(4)+
037174  106737  025414                    MFPS    @#PSWORD               ;SAVE PS
037200  122737  000000  025414            CMPB    #0,@#PSWORD            ;IS PS = 0
037206  001401                            BEQ     11$
037210  104000                            EMT                            ;PS IS WRONG
037212  022700  025252            11$:    CMP     #25252,%0              ;IS QUOTIENT = 25252
037216  001401                            BEQ     12$
037220  104000                            EMT                            ;QUOTIENT IS WRONG
037222  022701  000001            12$:    CMP     #1,%0+1                ;IS REMAINDER = 1
037226  001401                            BEQ     13$
037230  104000                            EMT                            ;WRONG REMAINDER
037232                            13$:
037232  005215                            INC     (R5)

                                  ;*******************************************************************
                                  ;TEST:306      DIV     0 52525 / @-(4) = 25252      REM = 1     PS = 0
                                  ;*******************************************************************
```

```
037234                                  TST306:
037234  012700  000000                          MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
037240  012701  052525                          MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
037244  071054                                  DIV     @-(4),%0        ;DIVIDE BY @-(4)
037246  106737  025414                          MFPS    @#PSWORD        ;SAVE PS
037252  122737  000000  025414                  CMPB    #0,@#PSWORD     ;IS PS = 0
037260  001401                                  BEQ     11$
037262  104000                                  EMT                     ;PS IS WRONG
037264  022700  025252                  11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
037270  001401                                  BEQ     12$
037272  104000                                  EMT                     ;QUOTIENT IS WRONG
037274  022701  000001                  12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
037300  001401                                  BEQ     13$
037302  104000                                  EMT                     ;WRONG REMAINDER
037304                                  13$:
037304  005215                                  INC     (R5)



                                        ;SPECIAL MULTIPLY DATA PATTERN TEST

037306  012701  177777                  TSTSPC: MOV     #-1,R1          ;MAKE R1 -1 SO WE KNOW INSTR. WAS MODIFIER
037312  012700  077700                          MOV     #77700,R0       ;SET UP TEST DATA
037316  070027  000001                          MUL     #1,R0           ;DO MULTIPLY INSTRUCTION
037322  022701  077700                          CMP     #77700,R1       ;CHECK LOW ORDER WORD
037326  001401                                  BEQ     1$
037330  104000                                  EMT                     ;LOW ORDER PRODUCT ERROR
037332  005700                          1$:     TST     R0              ;CHECK HIGH ORDER WORD
037334  001401                                  BEQ     EISEND
037336  104000                                  EMT                     ;HIGH ORDER PRODUCT ERROR
037340  000167  000026                  EISEND: JMP     MMUTST          ;JMP OVER GARBAGE AND GET TO MMU 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 "I" PAGE ADDRESS REGISTERS

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

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                        ;*KERNEL "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

        000006          KSP=    SP
        000006          USP=    SP
        177776          PSW=    PS
        000020          TBIT=   20
        000100          WBIT=   100
        000001          BIT0=   1
        000004          ERRVEC= 4

                        ;*ADDITIONAL DEFINITIONS
                        ;*



037344  000000          WASR6:  .WORD   0               ;USED TO STORE THE STACK POINTER AFTER A TRAP
037346  000000          TRAPPC: .WORD   0               ;USED TO STORE THE PC OF A TRAP OR ABORT
037350  000000          TRAPPS: .WORD   0               ;USED TO STORE THE PS OF A TRAP OR ABORT
037352  000000          WASSR0: .WORD   0               ;USED TO STORE CONTENTS OF SR0
037354  000000          WASSR2: .WORD   0               ;USED TO STORE CONTENTS OR SR2
037356  000000          TBITPS: .WORD   0               ;SAVES THE PSW THAT MAY HAVE ITS T-BIT ON
037360  000000          $TMP0:  .WORD   0               ;TEMPORARY STORAGE LOCATION
037362  000000          $TMP1:  .WORD   0               ;TEMPORARY STORAGE LOCATION
037364  000000          $TMP2:  .WORD   0               ;TEMPORARY STORAGE LOCATION
037366  000000          $TMP3:  .WORD   0               ;TEMPORARY STORAGE LOCATION
037370  000000          $TMP4:  .WORD   0               ;TEMPORARY STORAGE LOCATION
```

```

037372  012706  001000           MMUTST: MOV     #STBOT,KSP          ;INITIALIZE THE STACK POINTER
037376  012767  021356  140644           MOV     #T0250,MMVEC        ;LOAD MEMORY MANAGENT ROUTINE INTO VECTOR
037404  012767  000340  140640           MOV     #340,MMVEC+2        ;SET NEW PS TO PRIORITY LEVEL 7-KERNEL
037412  012767  000340  177736           MOV     #340,TBITPS         ;INITIALIZE LOG THAT HOLDS T-BIT PSW
037420  005067  140146                   CLR     SR0                 ;BE SURE MEM. MGMT IS OFF TO START WITH
037424  000244                           CLZ                         ;CLR THE Z BIT
037426  032777  000001  161750           BIT     #1,@SWR
037434  001402                           BEQ     1$
037436  000167  011076                   JMP     EXATST
037442  012737  050434  000030   1$:     MOV     #ERROR3,@#30        ;SET UP EMT VECTOR TO GO TO RIGHT ERROR CALL
037450  012737  000002  001004           MOV     #2,@#$TESTN         ;INCREMENT TEST NUMBER

```

```


                                  ;*****************************************************************************
                                  ;TEST 351        PSW PRIORITY BIT TEST
                                  ;*****************************************************************************
 037456                           TS351:
 037456  005000                           CLR     R0              ;INITIALIZE R0 WITH PRIORITY=0 DATA
 037460  005001                   2$:     CLR     R1              ;PREPARE R1 TO ACCEPT DATA READ
 037462  106400                           MTPS    R0              ;WRITE PRIORITY BITS IN THE PSW
 037464  106701                           MFPS    R1              ;READ BACK THE LOW BYTE OF PSW
 037466  042701  177437                   BIC     #177437,R1      ;MASK OFF EVERYTHING EXCEPT PRIORITY BITS
 037472  020001                           CMP     R0,R1           ;WAS CORRECT PRIORITY SET IN THE PSW?
 037474  001401                           BEQ     3$
 037476  104000                           EMT                     ;PRIORITY BITS SET WRONG IN PSW
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;'BR 2$" = 000770
 037500  062700  000040           3$:     ADD     #40,R0          ;CHANGE DATA TO NEXT PRIORITY
 037504  022700  000400                   CMP     #400,R0         ;HAVE PRIORITIES 0-7 ALL BEEN CHECKED?
 037510  001363                           BNE     2$              ;BRANCH IF NO

                                  ;*****************************************************************************
                                  ;TEST 352        PSW MODE BIT TEST
                                  ;*****************************************************************************
 037512                           TS352:
 037512  005000                           CLR     R0              ;INITIALIZE R0 WITH MODE BITS = 0000
 037514  005067  140256           2$:     CLR     PSW             ;INITIALIZE PSW
 037520  050067  140252                   BIS     R0,PSW          ;BIT SET THE PSW MODE BITS WITH R0
 037524  016701  140246                   MOV     PSW,R1          ;READ BACK THE CONTENTS OF THE PSW
 037530  042701  007777                   BIC     #007777,R1      ;MASK OFF EVERYTHING EXCEPT THE MODE BITS
 037534  020001                           CMP     R0,R1           ;WERE THE MODE BITS SET CORRECTLY?
 037536  001401                           BEQ     3$
 037540  104000                           EMT                     ;MODE BITS SET WRONG IN PSW
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;'BR 2$" = 000763
 037542  062700  010000           3$:     ADD     #10000,R0       ;CHANGE MODE BIT DATA
 037546  001362                           BNE     2$              ;BRANCH IF STILL MORE COMBINATIONS
 037550  005067  140222                   CLR     PSW             ;RESET PSW BEFORE LEAVING

                                  ;*****************************************************************************
                                  ;TEST 353        BYTE ADDRESSING TEST FOR PSW
                                  ;*****************************************************************************
 037554                           TS353:
 037554  005067  140216           2$:     CLR     PSW             ;CLEAR THE PSW
 037560  012700  000360                   MOV     #360,R0         ;PUT THE HIGH BYTE DATA INTO R0
 037564  110067  140207                   MOVB    R0,PSW+1        ;WRITE THE HIGH BYTE OF THE PSW
 037570  016701  140202                   MOV     PSW,R1          ;READ BACK THE ENTIRE PSW
 037574  042701  007437                   BIC     #007437,R1      ;MASK OFF THE T & CC BITS
 037600  000300                           SWAB    R0              ;GET DATA WRITTEN IN HIGH BYTE OF R0
 037602  020001                           CMP     R0,R1           ;WAS THE PSW WRITTEN TO CORRECTLY
 037604  001401                           BEQ     4$
 037606  104000                           EMT                     ;LOW BYTE EFFECTED BY WRITE TO HIGH BYTE OF PSW
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;'BR 2$" = 000760
```

```
037610  005067  140162          4$:     CLR     PSW             ;CLEAR THE PSW
037614  012700  000340                  MOV     #340,R0         ;PUT THE LOW BYTE DATA INTO R0
037620  110067  140152                  MOVB    R0,PSW          ;WRITE THE LOW BYTE OF THE PSW
037624  016701  140146                  MOV     PSW,R1          ;READ BACK THE ENTIRE PSW
037630  042701  007437                  BIC     #007437,R1      ;MASK OFF THE T&CC BITS
037634  020001                          CMP     R0,R1           ;WAS PSW WRITTEN TO CORRECTLY
037636  001401                          BEQ     TS354
037640  104000                          EMT                     ;HIGH BYTE EFFECTED BY WRITE TO LOW BYTE OF PSW
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 2$" = 000736

                                ;*******************************************************************************
                                ;TEST 354       TEST AND SETUP OF STACK POINTERS
                                ;*******************************************************************************
037642                          TS354:
037642  005067  140130                  CLR     PSW             ;GO TO KERNEL MODE
037646  012706  001000                  MOV     #KERSTK,KSP     ;SET KERNEL STACK POINTER TO 1100
037652  012767  140000  140116          MOV     #140000,PSW     ;GO TO USER MODE
037660  012706  000600                  MOV     #USESTK,USP     ;SET USER STACK POINTER TO 700
037664  005067  140106                  CLR     PSW             ;BACK TO KERNEL MODE
037670  022706  001000                  CMP     #KERSTK,KSP     ;IS KERNEL R6 STILL 1100?
037674  001401                          BEQ     TS355
037676  104000                          EMT                     ;KERNEL R6 CHANGED BY WRITING USER R6
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;000756

                                ;*******************************************************************************
                                ;*
                                ;*      THE NEXT FIVE (5) TESTS WILL TRY TO ADDRESS ALL OF THE
                                ;*      MEMORY MANAGEMENT REGISTERS (SR0,SR1,SR2,KERNEL & USER PAR/PDR'S).
                                ;*      EVERY TIME A REGISTER TIMES OUT ITS ADDRESS WILL BE REPORTED.
                                ;*      AT THE END OF EACH TEST A SUMMARY OF THE ADDRESSES THAT TIMED
                                ;*      OUT DURING THAT TEST IS GIVEN.  THE RESULTS OF "AND-ING" AND "OR-ING"
                                ;*      THEIR ADDRESSES IS GIVEN TO SHOW WHICH ADDRESS LINES MAY BE
                                ;*      STUCK AT 0 OR 1.  THE PAR/PDR ADDRESS AND KT MUX'S ARE THE
                                ;*      THINGS BEING CHECKED.
                                ;*
                                ;*******************************************************************************


                                ;*******************************************************************************
                                ;TEST 355       SR0,SR1,SR2,SR3 TIMEOUT TEST
                                ;*******************************************************************************
037700                          TS355:
037700  012700  177572                  MOV     #SR0,R0         ;LOAD R0 WITH ADDRESS OF FIRST REG.
037704  012701  000003                  MOV     #3,R1           ;LOAD R1 WITH THE LOOP COUNT
037710  005710                  2$:     TST     (R0)            ;TRY ADDRESSING A STATUS REGISTER
                                                                ;IF IT TIMES OUT GO TO 5$
037712  062700  000002          3$:     ADD     #2,R0           ;PUT NEXT ADDRESS IN R0
037716  077104                          SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
037720  005737  172516                  TST     @#172516        ;CHECK SR3 FOR RESPONSE

                                ;*******************************************************************************
                                ;TEST 356       KERNEL PAR'S TIMEOUT TEST
```

```
                                ;*******************************************************************************
037724                          TS356:

037724  012700  172340                  MOV     #KIPAR0,R0          ;LOAD R0 WITH ADDRESS OF FIRST REG.
037730  012701  000010                  MOV     #10,R1              ;LOAD R1 WITH LOOP COUNT (8)
037734  005710                  2$:     TST     (R0)                ;TRY ADDRESSING A KIPAR
                                                                    ;IF IT TIMES OUT, WILL GO TO 5$
037736  062700  000002          3$:     ADD     #2,R0               ;PUT NEXT KIPAR ADDRESS IN R0
037742  077104                          SOB     R1,2$               ;LOOP BACK TO 2$ UNTIL ALL TESTED

                                ;*******************************************************************************
                                ;TEST 357       KERNEL PDR'S TIMEOUT TEST
                                ;*******************************************************************************
037744                          TS357:

037744  012700  172300                  MOV     #KIPDR0,R0          ;LOAD R0 WITH ADDRESS OF FIRST REG.
037750  012701  000010                  MOV     #10,R1              ;LOAD R1 WITH LOOP COUNT (8)
037754  005710                  2$:     TST     (R0)                ;TRY ADDRESSING A KIPDR
                                                                    ;IF IT TIMES OUT, WILL GO TO 5$
037756  062700  000002          3$:     ADD     #2,R0               ;PUT NEXT KIPDR ADDRESS IN R0
037762  077104                          SOB     R1,2$               ;LOOP BACK TO 2$ UNTIL ALL TESTED

                                ;*******************************************************************************
                                ;TEST 360       USER PAR'S TIMEOUT TEST
                                ;*******************************************************************************
037764                          TS360:

037764  012700  177640                  MOV     #UIPAR0,R0          ;LOAD R0 WITH ADDRESS OF FIRST REG.
037770  012701  000010                  MOV     #10,R1              ;LOAD R1 WITH LOOP COUNT (8)
037774  005710                  2$:     TST     (R0)                ;TRY ADDRESSING A UIPAR
                                                                    ;IF IT TIMES OUT, WILL GO TO 5$
037776  062700  000002          3$:     ADD     #2,R0               ;PUT NEXT UIPAR ADDRESS IN R0
040002  077104                          SOB     R1,2$               ;LOOP BACK TO 2$ UNTIL ALL TESTED

                                ;*******************************************************************************
                                ;TEST 361       USER PDR'S TIMEOUT TEST
                                ;*******************************************************************************
040004                          TS361:

040004  012700  177600                  MOV     #UIPDR0,R0          ;LOAD R0 WITH ADDRESS OF FIRST REG.
040010  012701  000010                  MOV     #10,R1              ;LOAD R1 WITH LOOP COUNT (8)
040014  005710                  2$:     TST     (R0)                ;TRY ADDRESSING A UIPDR
                                                                    ;IF IT TIMES OUT, WILL GO TO 5$
040016  062700  000002          3$:     ADD     #2,R0               ;PUT NEXT UIPDR ADDRESS IN R0
040022  077104                          SOB     R1,2$               ;LOOP BACK TO 2$ UNTIL ALL TESTED

                                ;*******************************************************************************
                                ;TEST 362       SR0(15:13) BIT TEST & SR2 TEST
                                ;*******************************************************************************
040024                          TS362:

040024  012700  177572          1$:     MOV     #SR0,R0             ;LOAD ADDRESS OF SR0 INTO R0
040030  012710  160000                  MOV     #160000,(R0)        ;SET BITS <15:13> IN SR0 (ERROR BITS)
040034  000005                          RESET                       ;ISSUE AND ''INIT'' SIGNAL
040036  011001                          MOV     (R0),R1             ;READ SR0 INTO R1 TO SEE IF CLEAR
040040  001401                          BEQ     2$
```

```
040042  104000                          EMT                     ;SR0<15:13> NOT CLEARED BY A "RESET"
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 1$" = 000770
040044  016767  137526  177302  2$:     MOV     SR2,WASSR2      ;READ CONTENTS OF SR2
040052  012701  040044                  MOV     #2$,R1          ;LOAD EXPECTED CONTENTS INTO R1
040056  020167  177272                  CMP     R1,WASSR2       ;IS SR2 TRACKING?
040062  001401                          BEQ     3$
040064  104000                          EMT                     ;SR2 NOT "TRACKING" VIRTUAL ADDRESSES
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 2$" = 000767
040066  012701  100000          3$:     MOV     #100000,R1      ;PUT DATA TO BE WRITTEN IN R1
040072  012703  000003                  MOV     #3,R3           ;SETUP R3 AS A LOOP COUNTER
040076  005010                  4$:     CLR     (R0)            ;CLEAR SR0
040100  050110                  5$:     BIS     R1,(R0)         ;SET ONE OF THE ERROR BITS IN SR0
040102  011002                          MOV     (R0),R2         ;READ SR0 INTO R2
040104  020102                          CMP     R1,R2           ;DID RIGHT ERROR BIT GET SET?
040106  001401                          BEQ     6$
040110  104000                          EMT                     ;BITS WERE SET WRONG IN SR0
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 4$" = 000772
040112  012704  040100          6$:     MOV     #5$,R4          ;LOAD EXPECTED CONTENTS OF SR2 IN R4
040116  016767  137454  177230          MOV     SR2,WASSR2      ;READ SR2
040124  020467  177224                  CMP     R4,WASSR2       ;DID SR2 LOCK UP WHEN ERROR
                                                                ;BIT SET IN SR1?
040130  001401                          BEQ     7$
040132  104000                          EMT                     ;SR2 DID NOT LOCK UP
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 4$" = 000761
040134  006001                  7$:     ROR     R1              ;CHANGE DATA TO CHECK NEXT ERROR BIT
040136  077321                          SOB     R3,4$           ;LOOP BACK UNTIL <15:13> ALL TESTED
040140  005010                          CLR     (R0)            ;CLEAR SR0 BEFORE LEAVING

                                ;*****************************************************************
                                ;TEST 363       SR0 & PSW DUAL ADDRESSING TEST
                                ;*****************************************************************
040142                          TS363:

040142  005067  137630          1$:     CLR     PSW             ;CLEAR THE PSW
040146  005067  137420                  CLR     SR0             ;CLEAR STATUS REGISTER 0
040152  106427  000340                  MTPS    #340            ;SET PRIORITY 7 IN LOW BYTE OF PSW
040156  016700  137410                  MOV     SR0,R0          ;READ STATUS REGISTER 0
040162  001401                          BEQ     2$
040164  104000                          EMT                     ;SR0 EFFECTED BY A WRITE TO THE PSW
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 1$" = 000767
040166  005067  137400          2$:     CLR     SR0             ;BE SURE SR0 IS 0 BEFORE LEAVING
040172  005067  137600                  CLR     PSW             ;BE SURE PSW IS 0 BEFORE LEAVING

                                ;*****************************************************************
                                ;TEST 364       TEST THAT SR1 READS ALL ZEROS
                                ;*****************************************************************
```

```
040176                          TS364:
040176  012700  177777          1$:     MOV     #-1,R0          ;FILL R0 WITH ALL ONES
040202  016700  137366                  MOV     SR1,R0          ;READ SR1 INTO R0
040206  001401                          BEQ     2$
040210  104000                          EMT                     ;SR1 DID NOT READ ALL ZEROS
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;000772
040212  012767  177777  132276  2$:     MOV     #-1,SR3         ;TRY TO WRITE ONES TO SR3
040220  022767  000060  132270          CMP     #60,SR3         ;ONLY BITS <5:4> SHOULD BE ONES
040226  001401                          BEQ     3$
040230  104000                          EMT                     ;DIDN'T READ BACK A ''60''
040232  004567  010152          3$:     JSR     R5,CHKAPT
040236  000402                          BR      90$
040240  000005                          RESET                   ;CLEARS SR3
040242  000402                          BR      91$
040244  005067  132246          90$:    CLR     SR3
040250  005767  132242          91$:    TST     SR3             ;VERIFY THAT IT WAS CLEARED
040254                          4$:
040254  001401                          BEQ     TS365
040256  104000                          EMT                     ;SR3 DIDN'T READ ALL ZEROS


                                ;NOTE   F11 CHANGES INCLUDED CHECKING ALL BITS<15:0> OF PARS
                                ;       INSTEAD OF ONLY BITS<11:0>.

                                ;*******************************************************************************
                                ;TEST 365       BIT TEST OF KERNEL & USER PAR'S
                                ;*******************************************************************************
040260                          TS365:

040260  012700  172340          1$:     MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST PAR IN R0
040264  012703  000010          2$:     MOV     #10,R3          ;SETUP R3 TO COUNT 8 PAR'S
040270  005010                  3$:     CLR     (R0)            ;CLEAR THE PAR
040272  011001                          MOV     (R0),R1         ;READ THE PAR INTO R1
040274  001401                          BEQ     4$
040276  104000                          EMT                     ;PAR WOULD NOT CLEAR
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 3$" = 000774
040300  012704  077777          4$:     MOV     #077777,R4      ;LOAD "WALKING 0" TEST PATTERN IN R4
040304  005010                  5$:     CLR     (R0)            ;CLEAR THE PAR BEFORE LOADING DATA
040306  050410                          BIS     R4,(R0)         ;BIT SET THE TEST PATTERN INTO THE PAR
040310  011002                          MOV     (R0),R2         ;READ THE PAR INTO R2
040312  020402                          CMP     R4,R2           ;DOES DATA WRITTEN=DATA READ?
040314  001401                          BEQ     6$
040316  104000                          EMT                     ;PAR BITS DID NOT SET CORRECTLY
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 5$" = 000767
040320  000261                  6$:     SEC                     ;SET THE C-BIT FOR THE ROTATE INST.
040322  006004                          ROR     R4              ;ROTATE THE TEST PATTERN IN R4
040324  103767                          BCS     5$              ;BRANCH BACK IF MORE BITS TO TEST
040326  062700  000002                  ADD     #2,R0           ;GET NEXT PAR ADDRESS IN R0
040332  077322                          SOB     R3,3$           ;BRANCH BACK UNTIL ALL PAR'S TESTED
040334  022700  177660                  CMP     #UIPAR7+2,R0    ;HAVE USER PAR'S BEEN TESTED
```

```
040340  103003                          BHIS    TS366           ;GET TO NEXT TEST
040342  012700  177640                  MOV     #UIPAR0,R0      ;LOAD FIRST USER PAR ADDR. IN R0
040346  000746                          BR      2$              ;BRANCH BACK TO TEST USER PAR'S
                                                                ;LEAVE TEST WITH BITS <11:1>=1 IN ALL PAR'S
                        ;*******************************************************************************
                        ;TEST 366       BIT TEST OF KERNEL & USER PDR'S
                        ;*******************************************************************************
040350                  TS366:

040350  012700  172300  1$:     MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST PDR IN R0
040354  012703  000010  2$:     MOV     #10,R3          ;SETUP R3 TO COUNT 8 PDR'S
040360  005010          3$:     CLR     (R0)            ;CLEAR THE PDR
040362  011001                  MOV     (R0),R1         ;READ THE PDR INTO R1
040364  001401                  BEQ     4$
040366  104000                  EMT                     ;PDR WOULD NOT CLEAR
                                                        ;FOR TIGHTER SCOPE LOOP
                                                        ;REPLACE ERROR CALL WITH
                                                        ;"BR 3$" = 000774
040370  012704  077777  4$:     MOV     #077777,R4      ;LOAD "WALKING "0" TEST PATTERN IN R4
040374  005010          5$:     CLR     (R0)            ;CLEAR THE PDR BEFORE LOADING DATA
040376  010401                  MOV     R4,R1           ;LOAD DATA INTO R1
040400  042701  100361          BIC     #100361,R1      ;MASK UNUSED BITS OUT OF THE DATA
040404  050110                  BIS     R1,(R0)         ;BIT SET THE TEST PATTERN INTO THE PDR
040406  011002                  MOV     (R0),R2         ;READ THE PDR INTO R2
040410  020102                  CMP     R1,R2           ;DOES DATA WRITTEN=DATA READ?
040412  001401                  BEQ     6$
040414  104000                  EMT                     ;PDR BITS DID NOT SET CORRECTLY
                                                        ;FOR TIGHTER SCOPE LOOP
                                                        ;REPLACE ERROR CALL WITH
                                                        ;"BR 5$" = 000767
040416  000261          6$:     SEC                     ;SET THE C-BIT FOR THE ROTATE INST.
040420  006004                  ROR     R4              ;ROTATE THE TEST PATTERN IN R4
040422  103764                  BCS     5$              ;BRANCH BACK IF MORE BITS TO TEST
040424  062700  000002          ADD     #2,R0           ;GET NEXT PDR ADDRESS IN R0
040430  077325                  SOB     R3,3$           ;BRANCH BACK UNTIL ALL PDR'S TESTED
040432  022700  177620          CMP     #UIPDR7+2,R0    ;HAVE USER PDR'S BEEN TESTED?
040436  103003                  BHIS    TS367           ;GET TO NEXT TEST
040440  012700  177600          MOV     #UIPDR0,R0      ;LOAD FIRST USER PDR ADDR. IN R0
040444  000743                  BR      2$              ;BRANCH BACK TO TEST USER PDR'S
                                                        ;LEAVE TEST WITH ALL WRITEABLE BITS IN
                                                        ;ALL PDR'S = 1

                        ;*******************************************************************************
                        ;TEST 367       TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PAR'S
                        ;*******************************************************************************
040446                  TS367:

040446  012700  172340  1$:     MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST PAR INTO R0
040452  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER TO DO 8 PAR'S
040456  012701  177777  3$:     MOV     #-1,R1          ;LOAD TEST PATTERN INTO R1
040462  005010                  CLR     (R0)            ;CLEAR THE PAR
040464  110110                  MOVB    R1,(R0)         ;WRITE 1'S TO THE LOW BYTE OF THE PAR
040466  011002                  MOV     (R0),R2         ;READ THE ENTIRE PAR INTO R2
040470  042701  177400          BIC     #177400,R1      ;MASK HIGH BYTE & UNUSED BITS OUT OF THE DATA
040474  020102                  CMP     R1,R2           ;WAS ONLY THE LOW BYTE WRITTEN TO
040476  001401                  BEQ     5$
```

```
040500  104000                         EMT                          ;HIGH BYTE EFFECTED BY WRITING LOW BYTE IN PAR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 3$'' = 000766
040502  005010                 5$:     CLR     (R0)                 ;CLEAR THE PAR
040504  012701  177777                 MOV     #-1,R1               ;LOAD TEST, PATTERN INTO R1
040510  110160  000001                 MOVB    R1,1(R0)             ;WRITE 1'S TO THE HIGH BYTE OF THE PAR
040514  011002                         MOV     (R0),R2              ;READ THE ENTIRE PAR INTO R2
                                                                    ;F11 CHANGE      WAS #170377
040516  042701  000377                 BIC     #000377,R1           ;MASK LOW BYTE & UNUSED BITS OUT OF DATA
040522  020102                         CMP     R1,R2                ;WAS ONLY THE HIGH BYTE WRITTEN TO?
040524  001401                         BEQ     6$
040526  104000                         EMT                          ;LOW BYTE EFFECTED BY WRITING HIGH BYTE IN PAR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 5'' = 000765
040530  062700  000002         6$:     ADD     #2,R0                ;PUT ADDRESS OF NEXT PAR IN R0
040534  077330                         SOB     R3,3$                ;BRANCH BACK UNTIL 8 PAR'S TESTED
040536  022700  177660                 CMP     #UIPAR7+2,R0         ;HAVE USER PAR'S BEEN TESTED
040542  103003                         BHIS    TS370                ;GET TO NEXT TEST
040544  012700  177640                 MOV     #UIPAR0,R0           ;LOAD ADDRESS OF FIRST USER PAR IN R0
040550  000742                         BR      3$                   ;BRANCH BACK TO TEST USER PAR'S

                               ;*****************************************************************************
                               ;TEST 370        TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PDR'S
                               ;*****************************************************************************
040552                         TS370:

040552  012700  172300         1$:     MOV     #KIPDR0,R0           ;LOAD ADDRESS OF FIRST PDR INTO R0
040556  012703  000010                 MOV     #10,R3               ;LOAD LOOP COUNTER TO DO 8 PDR'S
040562  012701  177777         3$:     MOV     #-1,R1               ;LOAD TEST PATTERN INTO R1
040566  005010                         CLR     (R0)                 ;CLEAR THE PDR
040570  110110                         MOVB    R1,(R0)              ;WRITE 1'S TO THE LOW BYTE OF THE PDR
040572  011002                         MOV     (R0),R2              ;READ THE ENTIRE PDR INTO R2
040574  042701  177761                 BIC     #177761,R1           ;MASK HIGH BYTE & UNUSED BITS OUT OF DATA
040600  020102                         CMP     R1,R2                ;WAS ONLY THE LOW BYTE WRITTEN TO?
040602  001401                         BEQ     5$
040604  104000                         EMT                          ;HIGH BYTE EFFECTED BY WRITING LOW BYTE IN PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 3$'' = 000766
040606  005010                 5$:     CLR     (R0)                 ;CLEAR THE PDR
040610  012701  177777                 MOV     #-1,R1               ;LOAD TEST PATTERN INTO R1
040614  110160  000001                 MOVB    R1,1(R0)             ;WRITE 1'S TO THE HIGH BYTE OF THE PDR
040620  011002                         MOV     (R0),R2              ;READ THE ENTIRE PDR INTO R2
040622  042701  100377                 BIC     #100377,R1           ;MASK LOW BYTE & UNUSED BITS OUT OF DATA
040626  020102                         CMP     R1,R2                ;WAS ONLY THE HIGH BYTE WRITTEN TO?
040630  001401                         BEQ     6$
040632  104000                         EMT                          ;LOW BYTE EFFECTED BY WRITING HIGH BYTE IN PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 5$'' = 000765
040634  062700  000002         6$:     ADD     #2,R0                ;PUT ADDRESS OF NEXT PDR IN R0
040640  077330                         SOB     R3,3$                ;BRANCH BACK UNTIL 8 PDR'S TESTED
040642  022700  177620                 CMP     #UIPDR7+2,R0         ;HAVE USER PDR'S BEEN TESTED?
040646  103003                         BHIS    TS371                ;GET TO NEXT TEST
```

```
040650  012700  177600                    MOV     #UIPDR0,R0       ;LOAD ADDRESS OF FIRST USER PDR IN R0
040654  000742                            BR      3$               ;BRANCH BACK TO TEST USER PDR'S

                                  ;*************************************************************************
                                  ;TEST 371        PAR-PDR DUAL ADDRESSING TEST
                                  ;*************************************************************************
        040656                    TS371:

040656  012703  000010                    MOV     #10,R3           ;LOAD LOOP COUNTER WITH AN 8
040662  012700  172300                    MOV     #KIPDR0,R0       ;LOAD ADDRESS OF FIRST KERNEL PDR AND R0
040666  004767  007242                    JSR     PC,SETREG        ;SET ALL BITS IN ALL PAR'S IN PDR'S
040672  012706  001000            2$:     MOV     #KERSTK,KSP      ;SETUP STACK POINTER
040676  005010                            CLR     (R0)             ;CLEAR ONE OF THE KERNEL PDR'S
040700  004767  007322                    JSR     PC,CMPREG        ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
040704  012720  177777                    MOV     #-1,(R0)+        ;RESTORE ALL ONES, AND SETUP FOR NEXT PDR
040710  077310                            SOB     R3,2$            ;LOOP TO 2$ UNTIL ALL KERNEL PDR'S CHECKED
040712  012703  000010                    MOV     #10,R3           ;LOAD LOOP COUNTER WITH AN 8
040716  012700  172340                    MOV     #KIPAR0,R0       ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
040722  012706  001000            3$:     MOV     #KERSTK,KSP      ;SETUP STACK POINTER
040726  005010                            CLR     (R0)             ;CLEAR ONE OF THE KERNEL PAR'S
040730  004767  007272                    JSR     PC,CMPREG        ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
040734  012720  177777                    MOV     #-1,(R0)+        ;RESTORE ALL ONES, AND SETUP FOR NEXT PAR
040740  077310                            SOB     R3,3$            ;LOOP TO 3$ UNTIL ALL KERNEL PAR'S CHECKED
040742  012703  000010                    MOV     #10,R3           ;LOAD LOOP COUNTER WITH AN 8
040746  012700  177600                    MOV     #UIPDR0,R0       ;LOAD ADDRESS OF FIRST USER PDR IN R0
040752  012706  001000            4$:     MOV     #KERSTK,KSP      ;SETUP STACK POINTER
040756  005010                            CLR     (R0)             ;CLEAR ONE OF THE USER PDR'S
040760  004767  007242                    JSR     PC,CMPREG        ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
040764  012720  177777                    MOV     #-1,(R0)+        ;RESTORE ALL ONES, AND SETUP FOR NEXT UPDR
040770  077310                            SOB     R3,4$            ;LOOP TO 4$ UNTIL ALL USER PDR'S CHECKED
040772  012703  000010                    MOV     #10,R3           ;LOAD LOOP COUNTER WITH AN 8
040776  012700  177640                    MOV     #UIPAR0,R0       ;LOAD ADDRESS OF FIRST USER PAR IN R0
041002  012706  001000            5$:     MOV     #KERSTK,KSP      ;SETUP STACK POINTER
041006  005010                            CLR     (R0)             ;CLEAR ONE OF THE USER PAR'S
041010  004767  007212                    JSR     PC,CMPREG        ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
041014  012720  177777                    MOV     #-1,(R0)+        ;RESTORE ALL ONES, AND SETUP FOR NEXT UPAR
041020  077310                            SOB     R3,5$            ;LOOP TO 5$ UNTIL ALL USER PAR'S CHECKED

                                  ;*************************************************************************
                                  ;TEST 372        TEST THAT PAR-PDR'S NOT AFFECTED BY RESET
                                  ;*************************************************************************
        041022                    TS372:


041022  032737  000001  001020            BIT     #1, @#$ENV       ;ARE WE RUNNING UNDER APT
041030  001403                            BEQ     70$              ;IF NO THEN DO TEST
041032  005737  001006                    TST     @#$PASS          ;IS THIS FIRST PASS
041036  001063                            BNE     TS373            ;IF NO THEN SHIP TO NEXT TEST
041040                            70$:
041040  004767  007070            1$:     JSR     PC,SETREG        ;SET ALL BITS IN ALL PAR'S AND PDR'S
041044  000005                            RESET                    ;ISSUE AN ''INIT'' BY EXECUTING A RESET
041046  012700  172300            10$:    MOV     #KIPDR0,R0       ;LOAD ADDRESS OF FIRST KERNEL PDR IN R0
041052  012704  000010                    MOV     #10,R4           ;LOAD LOOP COUNTER WITH AN 8
041056  011001                    2$:     MOV     (R0),R1          ;READ A KERNEL PDR INTO R1
041060  022701  077416                    CMP     #77416,R1        ;ARE ALL THE BITS STILL SET?
041064  001401                            BEQ     3$
```

```
041066  104000                          EMT                     ;KERNEL PDR AFFECTED BY A RESET
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 2$' = 000773
041070  062700  000002          3$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT KERNEL PDR
041074  077410                          SOB     R4,2$           ;LOOP TO 2$ UNTIL ALL KERNEL PDR'S CHECKED
041076  012700  172340                  MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
041102  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
041106  011001                  4$:     MOV     (R0),R1         ;READ A KERNEL PAR INTO R1
                                                                ;****F11 CHANGE****      WAS #7777
041110  022701  177777                  CMP     #177777,R1      ;ARE ALL THE BITS STILL SET?
041114  001401                          BEQ     5$
041116  104000                          EMT                     ;KERNEL PAR AFFECTED BY A RESET
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 4$' = 000773
041120  062700  000002          5$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT KERNEL PAR
041124  077410                          SOB     R4,4$           ;LOOP TO 4$ UNTIL ALL KERNEL PAR'S CHECKED
041126  012700  177600                  MOV     #UIPDR0,R0      ;LOAD ADDRESS OF FIRST USER PDR IN R0
041132  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
041136  011001                  6$:     MOV     (R0),R1         ;READ A USER PDR INTO R1
041140  022701  077416                  CMP     #77416,R1       ;ARE ALL THE BITS STILL SET?
041144  001401                          BEQ     7$
041146  104000                          EMT                     ;USER PDR AFFECTED BY A RESET
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 6$' = 000773
041150  062700  000002          7$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT USER PDR
041154  077410                          SOB     R4,6$           ;LOOP TO 6$ UNTIL ALL USER PDR'S CHECKED

041156  012700  177640                  MOV     #UIPAR0,R0      ;LOAD ADDRESS OF FIRST USER PAR IN R0
041162  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
041166  011001                  8$:     MOV     (R0),R1         ;READ A USER PAR INTO R1
                                                                ;****F11 CHANGE****      WAS #7777
041170  022701  177777                  CMP     #177777,R1      ;ARE ALL THE BITS STILL SET?
041174  001401                          BEQ     9$
041176  104000                          EMT                     ;USER PAR AFFECTED BY A RESET
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 8$' = 000773
041200  062700  000002          9$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT USER PAR
041204  077410                          SOB     R4,8$           ;LOOP TO 8$ UNTIL ALL USER PAR'S CHECKED
```

```
                                    ;*********************************************************************
                                    ;TEST 373         RELOCATION & ADDER TEST (NO CARRIES)
                                    ;*********************************************************************
041206                              TS373:

041206  012700  172340              1$:     MOV     #KIPAR0,R0          ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
041212  005001                              CLR     R1                  ;CLEAR R1
041214  012702  000007                      MOV     #7,R2               ;LOAD LOOP COUNTER WITH A 7
041220  010120                      2$:     MOV     R1,(R0)+            ;MAP KERNEL PAR'S TO PAGES 0-6 (4K EACH)
041222  062701  000200                      ADD     #200,R1
041226  077204                              SOB     R2,2$               ;LOOP UNTIL KIPAR0 - KIPAR6 ARE LOADED
041230  012710  177600                      MOV     #177600,(R0)        ;MAP KIPAR7 TO THE I/O PAGE
041234  012700  172300                      MOV     #KIPDR0,R0          ;LOAD ADDRESS OF FIRST KERNEL PDR IN R0
041240  012701  077406                      MOV     #77406,R1           ;LOAD PDR DATA INTO R1
041244  012702  000010                      MOV     #10,R2              ;LOAD LOOP COUNTER WITH AN 8
041250  010120                      3$:     MOV     R1,(R0)+            ;MAP ALL 8 PAGES 128 BLOCKS, UPWARD
041252  077202                              SOB     R2,3$               ;       EXPANDABLE, READ/WRITE

041254  012700  067776              4$:     MOV     #67776,R0           ;LOAD PHYSICAL ADDR. PBA INTO R0
041260  012701  107776                      MOV     #107776,R1          ;LOAD VIRTUAL ADDR. VBA INTO R1
041264  012702  125250                      MOV     #125250,R2          ;LOAD TEST PATTERN INTO R2
041270  012704  000600                      MOV     #600,R4 ;LOAD R4 WITH PAR VALUE
041274  010467  131050                      MOV     R4,KIPAR4           ;LOAD KERNEL PAR 4 BITS <11:00>
041300  011067  176054                      MOV     (R0),$TMP0          ;SAVE CONTENTS AT TEST LOCATION
041304  005067  131206                      CLR     SR3                 ;SET UP FOR 18-BIT ADDRESSING
041310  052767  000001  136254              BIS     #BIT0,SR0           ;TURN ON 'RELOCATION'
041316  010211                              MOV     R2,(R1)             ;LOAD 125250 USING ADDER  (PAR4 + VIRT ADDR.)
041320  005067  136246                      CLR     SR0                 ;TURN OFF MEMORY MGMT.
041324  011003                              MOV     (R0),R3             ;READ 125250 BACK WITHOUT USING MEM. MGMT.
041326  016710  176026                      MOV     $TMP0,(R0)          ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
041332  020203                              CMP     R2,R3               ;WAS SAME PATTERN READ BACK THAT WAS
                                                                        ;WRITTEN USING 'DEST-ONLY-RELOC.'?
041334  001401                              BEQ     5$
041336  104000                              EMT                         ;TEST LOCATION DID NOT HAVE PATTERN
                                                                        ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                                        ;APPARENTLY PHYSICAL ADDR. WAS
                                                                        ;FORMED WRONG BY ADDERS USING
                                                                        ;THE VIRTUAL ADDR. AND KIPAR4
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 4$" = 000742
041340                              5$:
041340  012700  067776              6$:     MOV     #67776,R0           ;LOAD PHYSICAL ADDR. PBA INTO R0
041344  012701  102576                      MOV     #102576,R1          ;LOAD VIRTUAL ADDR. VBA INTO R1
041350  012702  125251                      MOV     #125251,R2          ;LOAD TEST PATTERN INTO R2
041354  012704  000652                      MOV     #652,R4 ;LOAD R4 WITH PAR VALUE
041360  010467  130764                      MOV     R4,KIPAR4           ;LOAD KERNEL PAR 4 BITS <11:00>
```

```
041364  011067  175770                  MOV     (R0),$TMP0          ;SAVE CONTENTS AT TEST LOCATION
041370  005067  131122                  CLR     SR3                 ;SET UP FOR 18-BIT ADDRESSING
041374  052767  000001  136170          BIS     #BIT0,SR0           ;TURN ON "RELOCATION"
041402  010211                          MOV     R2,(R1)             ;LOAD 125251 USING ADDER  (PAR4 + VIRT ADDR.)
041404  005067  136162                  CLR     SR0                 ;TURN OFF MEMORY MGMT.
041410  011003                          MOV     (R0),R3             ;READ 125251 BACK WITHOUT USING MEM. MGMT.
041412  016710  175742                  MOV     $TMP0,(R0)          ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
041416  020203                          CMP     R2,R3               ;WAS SAME PATTERN READ BACK THAT WAS
                                                                    ;WRITTEN USING "DEST-ONLY-RELOC."?
041420  001401                          BEQ     7$
041422  104000                          EMT                         ;TEST LOCATION DID NOT HAVE PATTERN
                                                                    ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                                    ;APPARENTLY PHYSICAL ADDR. WAS
                                                                    ;FORMED WRONG BY ADDERS USING
                                                                    ;THE VIRTUAL ADDR. AND KIPAR4
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 6$" = 000742
041424                          7$:
041424  012700  067776          8$:     MOV     #67776,R0           ;LOAD PHYSICAL ADDR. PBA INTO R0
041430  012701  105276                  MOV     #105276,R1          ;LOAD VIRTUAL ADDR. VBA INTO R1
041434  012702  125252                  MOV     #125252,R2          ;LOAD TEST PATTERN INTO R2
041440  012704  000625                  MOV     #625,R4 ;LOAD R4 WITH PAR VALUE
041444  010467  130700                  MOV     R4,KIPAR4           ;LOAD KERNEL PAR 4 BITS <11:00>
041450  011067  175704                  MOV     (R0),$TMP0          ;SAVE CONTENTS AT TEST LOCATION
041454  052767  000020  131034          BIS     #BIT4,SR3           ;SET UP FOR 22-BIT ADDRESSING
041462  052767  000001  136102          BIS     #BIT0,SR0           ;TURN ON "RELOCATION"
041470  010211                          MOV     R2,(R1)             ;LOAD 125252 USING ADDER  (PAR4 + VIRT ADDR.)
041472  005067  136074                  CLR     SR0                 ;TURN OFF MEMORY MGMT.
041476  011003                          MOV     (R0),R3             ;READ 125252 BACK WITHOUT USING MEM. MGMT.
041500  016710  175654                  MOV     $TMP0,(R0)          ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
041504  020203                          CMP     R2,R3               ;WAS SAME PATTERN READ BACK THAT WAS
                                                                    ;WRITTEN USING "DEST-ONLY-RELOC."?
041506  001401                          BEQ     9$
041510  104000                          EMT                         ;TEST LOCATION DID NOT HAVE PATTERN
                                                                    ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                                    ;APPARENTLY PHYSICAL ADDR. WAS
                                                                    ;FORMED WRONG BY ADDERS USING
                                                                    ;THE VIRTUAL ADDR. AND KIPAR4
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 8$" = 000742
041512                          9$:

041512  012700  177776                  MOV     #PSW,R0             ;LOAD PHYS. ADDR. OF PSW INTO R0
041516  012701  100076                  MOV     #100076,R1          ;LOAD VIRTUAL ADDR. FOR PSW INTO R1
041522  012702  030340                  MOV     #030340,R2          ;LOAD DATA FOR PSW IN R2
041526  012704  007777                  MOV     #7777,R4            ;LOAD R4 WITH PAR VALUE
041532  010467  130612                  MOV     R4,KIPAR4           ;LOAD KERNEL PAR 4 BITS <11:00>
041536  005010                          CLR     (R0)                ;CLEAR THE PSW
041540  005067  130752                  CLR     SR3                 ;SET UP FOR 18-BIT ADDRESSING
041544  052767  000001  136020          BIS     #BIT0,SR0           ;TURN ON "MEMORY MANAGEMENT"
041552  010211                          MOV     R2,(R1)             ;LOAD PSW USING ADDER (PAR4 + VIRT ADDR.)
041554  005067  136012                  CLR     SR0                 ;TURN OFF MEM. MGMT (SR0=0)
041560  011003                          MOV     (R0),R3             ;READ PSW BACK WITHOUT USING MEM. MGMT.
041562  005010                          CLR     (R0)                ;CLEAR THE PSW
```

```
041564  042703  000037                  BIC     #37,R3          ;MASK T-BIT & CC BITS OUT OF DATA READ
041570  020203                          CMP     R2,R3           ;WAS PSW WRITTEN?
041572  001401                          BEQ     11$
041574  104000                          EMT                     ;PSW DID NOT HAVE DATA THAT IT SHOULD HAVE,
                                                                ;APPARENTLY PHYS. ADDR. OF PSW WAS
                                                                ;NOT FORMED BY ADDERS USING THE
                                                                ;VIRTUAL ADDR. AND KIPAR4
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 10$'' = 000742
041576  012700  177776          11$:    MOV     #PSW,R0         ;LOAD PHYS. ADDR. OF PSW INTO R0
041602  012701  117776                  MOV     #117776,R1      ;LOAD VIRTUAL ADDR. FOR PSW INTO R1
041606  012702  030240                  MOV     #030240,R2      ;LOAD DATA FOR PSW IN R2
041612  012704  177600                  MOV     #177600,R4      ;LOAD R4 WITH PAR VALUE
041616  010467  130526                  MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
041622  052767  000020  130666          BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
041630  052767  000001  135734          BIS     #BIT0,SR0       ;TURN ON 'MEMORY MANAGEMENT''
041636  010211                          MOV     R2,(R1)         ;LOAD PSW USING ADDER (PAR4 + VIRT. ADDR.)
041640  005067  135726                  CLR     SR0             ;TURN OFF MEM. MGMT (SR0=0)
041644  011003                          MOV     (R0),R3         ;READ PSW BACK WITHOUT USING MEM. MGMT.
041646  005010                          CLR     (R0)            ;CLEAR THE PSW
041650  042703  000037                  BIC     #37,R3          ;MASK T-BIT & CC BITS OUT OF DATA READ
041654  020203                          CMP     R2,R3           ;WAS PSW WRITTEN WHILE IN MAINT. MODE?
041656  001401                          BEQ     TS374
041660  104000                          EMT                     ;PSW DID NOT HAVE DATA THAT IT SHOULD
                                                                ;HAVE, APPARENTLY PHYS. ADDR. OF PSW WAS
                                                                ;NOT FORMED BY ADDERS USING THE
                                                                ;VIRTUAL ADDR. AND KIPAR4
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 11$'' = 000743

                                ;*******************************************************************************
                                ;TEST 374       RELOCATION & ADDER TEST (WITH CARRIES)
                                ;*******************************************************************************
041662                          TS374:

041662                          1$:                             ;KERNEL PAR'S AND PDR'S HAVE BEEN
                                                                ;SETUP BY THE PREVIOUS TEST
041662  012700  066476          2$:     MOV     #66476,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
041666  012701  114376                  MOV     #114376,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
041672  012702  125253                  MOV     #125253,R2      ;LOAD TEST PATTERN INTO R2
041676  012704  000521                  MOV     #521,R4 ;LOAD R4 WITH PAR VALUE
041702  010467  130442                  MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
041706  011067  175446                  MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
041712  052767  000020  130576          BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
041720  052767  000001  135644          BIS     #BIT0,SR0       ;TURN ON 'RELOCATION''
041726  010211                          MOV     R2,(R1)         ;LOAD 125253 USING ADDER  (PAR4 + VIRT ADDR.)
041730  005067  135636                  CLR     SR0             ;TURN OFF MEMORY MGMT.
041734  011003                          MOV     (R0),R3         ;READ 125253 BACK WITHOUT USING MEM. MGMT.
041736  016710  175416                  MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
041742  020203                          CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
                                                                ;WRITTEN USING 'DEST-ONLY-RELOC.''?
041744  001401                          BEQ     3$
041746  104000                          EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
                                                                ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
```

```
                                                        ;APPARENTLY PHYSICAL ADDR. WAS
                                                        ;FORMED WRONG BY ADDERS USING
                                                        ;THE VIRTUAL ADDR. AND KIPAR4
                                                        ;FOR TIGHTER SCOPE LOOP
                                                        ;REPLACE ERROR CALL WITH
                                                        ;"BR 2$" = 000742
041750                          3$:
041750 012700 062276            4$:   MOV   #62276,R0     ;LOAD PHYSICAL ADDR. PBA INTO R0
041754 012701 107376                  MOV   #107376,R1    ;LOAD VIRTUAL ADDR. VBA INTO R1
041760 012702 125254                  MOV   #125254,R2    ;LOAD TEST PATTERN INTO R2
041764 012704 000527                  MOV   #527,R4 ;LOAD R4 WITH PAR VALUE
041770 010467 130354                  MOV   R4,KIPAR4     ;LOAD KERNEL PAR 4 BITS <11:00>
041774 011067 175360                  MOV   (R0),$TMPO    ;SAVE CONTENTS AT TEST LOCATION
042000 052767 000020 130510           BIS   #BIT4,SR3     ;SET UP FOR 22-BIT ADDRESSING
042006 052767 000001 135556           BIS   #BIT0,SR0     ;TURN ON 'RELOCATION'
042014 010211                         MOV   R2,(R1)       ;LOAD 125254 USING ADDER  (PAR4 + VIRT ADDR.)
042016 005067 135550                  CLR   SR0           ;TURN OFF MEMORY MGMT.
042022 011003                         MOV   (R0),R3       ;READ 125254 BACK WITHOUT USING MEM. MGMT.
042024 016710 175330                  MOV   $TMPO,(R0)    ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
042030 020203                         CMP   R2,R3         ;WAS SAME PATTERN READ BACK THAT WAS
                                                          ;WRITTEN USING 'DEST-ONLY-RELOC.'?
042032 001401                         BEQ   5$
042034 104000                         EMT                 ;TEST LOCATION DID NOT HAVE PATTERN
                                                          ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                          ;APPARENTLY PHYSICAL ADDR. WAS
                                                          ;FORMED WRONG BY ADDERS USING
                                                          ;THE VIRTUAL ADDR. AND KIPAR4
                                                          ;FOR TIGHTER SCOPE LOOP
                                                          ;REPLACE ERROR CALL WITH
                                                          ;"BR 4$" = 000742
042036                          5$:
042036 012700 062076            6$:   MOV   #62076,R0     ;LOAD PHYSICAL ADDR. PBA INTO R0
042042 012701 104576                  MOV   #104576,R1    ;LOAD VIRTUAL ADDR. VBA INTO R1
042046 012702 125255                  MOV   #125255,R2    ;LOAD TEST PATTERN INTO R2
042052 012704 000553                  MOV   #553,R4 ;LOAD R4 WITH PAR VALUE
042056 010467 130266                  MOV   R4,KIPAR4     ;LOAD KERNEL PAR 4 BITS <11:00>
042062 011067 175272                  MOV   (R0),$TMPO    ;SAVE CONTENTS AT TEST LOCATION
042066 052767 000020 130422           BIS   #BIT4,SR3     ;SET UP FOR 22-BIT ADDRESSING
042074 052767 000001 135470           BIS   #BIT0,SR0     ;TURN ON 'RELOCATION'
042102 010211                         MOV   R2,(R1)       ;LOAD 125255 USING ADDER  (PAR4 + VIRT ADDR.)
042104 005067 135462                  CLR   SR0           ;TURN OFF MEMORY MGMT.
042110 011003                         MOV   (R0),R3       ;READ 125255 BACK WITHOUT USING MEM. MGMT.
042112 016710 175242                  MOV   $TMPO,(R0)    ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
042116 020203                         CMP   R2,R3         ;WAS SAME PATTERN READ BACK THAT WAS
                                                          ;WRITTEN USING 'DEST-ONLY-RELOC.'?
042120 001401                         BEQ   7$
042122 104000                         EMT                 ;TEST LOCATION DID NOT HAVE PATTERN
                                                          ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                          ;APPARENTLY PHYSICAL ADDR. WAS
                                                          ;FORMED WRONG BY ADDERS USING
                                                          ;THE VIRTUAL ADDR. AND KIPAR4
                                                          ;FOR TIGHTER SCOPE LOOP
                                                          ;REPLACE ERROR CALL WITH
                                                          ;"BR 6$" = 000742
042124                          7$:
042124 012700 000000            8$:   MOV   #000000,R0    ;LOAD PHYSICAL ADDR. PBA INTO R0
```

```
042130  012701  111400                 MOV     #111400,R1        ;LOAD VIRTUAL ADDR. VBA INTO R1
042134  012702  125256                 MOV     #125256,R2        ;LOAD TEST PATTERN INTO R2
042140  012704  177664                 MOV     #177664,R4        ;LOAD R4 WITH PAR VALUE
042144  010467  130200                 MOV     R4,KIPAR4         ;LOAD KERNEL PAR 4 BITS <11:00>
042150  011067  175204                 MOV     (R0),$TMPO        ;SAVE CONTENTS AT TEST LOCATION
042154  052767  000020  130334         BIS     #BIT4,SR3         ;SET UP FOR 22-BIT ADDRESSING
042162  052767  000001  135402         BIS     #BIT0,SR0         ;TURN ON 'RELOCATION'
042170  010211                         MOV     R2,(R1)           ;LOAD 125256 USING ADDER  (PAR4 + VIRT ADDR.)
042172  005067  135374                 CLR     SR0               ;TURN OFF MEMORY MGMT.
042176  011003                         MOV     (R0),R3           ;READ 125256 BACK WITHOUT USING MEM. MGMT.
042200  016710  175154                 MOV     $TMPO,(R0)        ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
042204  020203                         CMP     R2,R3             ;WAS SAME PATTERN READ BACK THAT WAS
                                                                 ;WRITTEN USING 'DEST-ONLY-RELOC.'?
042206  001401                         BEQ     9$
042210  104000                         EMT                       ;TEST LOCATION DID NOT HAVE PATTERN
                                                                 ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                                 ;APPARENTLY PHYSICAL ADDR. WAS
                                                                 ;FORMED WRONG BY ADDERS USING
                                                                 ;THE VIRTUAL ADDR. AND KIPAR4
                                                                 ;FOR TIGHTER SCOPE LOOP
                                                                 ;REPLACE ERROR CALL WITH
                                                                 ;''BR 8$'' = 000742
042212                          9$:

                                ;*******************************************************************************
                                ;TEST 375          READ AND WRITE WHILE IN RELOCATE MODE
                                ;*******************************************************************************
042212                          TS375:

042212  005067  135560          1$:    CLR     PSW               ;START IN KERNEL MODE
042216  012704  001377                 MOV     #1377,R4          ;LOAD R4 WITH VALUE FOR PAR4
042222  012705  001400                 MOV     #1400,R5          ;LOAD R5 WITH VALUE FOR PAR5
042226  010467  130116                 MOV     R4,KIPAR4         ;LOAD KERNEL PAR4
042232  010567  130114                 MOV     R5,KIPAR5         ;LOAD KERNEL PAR5
042236  012700  177640                 MOV     #UIPARO,R0        ;LOAD ADDRESS OF FIRST USER PAR IN R0
042242  005001                         CLR     R1                ;CLEAR R1
042244  012702  000007                 MOV     #7,R2             ;LOAD LOOP COUNTER WITH A 7
042250  010120                  2$:    MOV     R1,(R0)+          ;MAP USER PAR'S TO PAGES 0-6 (4K EACH)
042252  062701  000200                 ADD     #200,R1
042256  077204                         SOB     R2,2$             ;LOOP UNTIL UIPARO-UIPAR6 ARE LOADED
042260  012710  177600                 MOV     #177600,(R0)      ;MAP USER PAR7 TO THE I/O PAGE
042264  012700  177600                 MOV     #UIPDRO,R0        ;LOAD ADDRESS OF FIRST USER PDR IN R0
042270  012701  077406                 MOV     #77406,R1         ;LOAD PDR DATA INTO R1
042274  012702  000010                 MOV     #10,R2            ;LOAD LOOP COUNTER WITH AN 8
042300  010120                  3$:    MOV     R1,(R0)+          ;MAP ALL 8 PAGES 128 BLOCKS, UPWARD
042302  077202                         SOB     R2,3$             ;        EXPANDABLE, READ/WRITE
042304  012767  042550  135736         MOV     #8$,MMVEC         ;SET M. M. TRAP VECTOR TO 8$
042312  052767  000020  130176         BIS     #BIT4,SR3         ;SET UP FOR 22-BIT ADDRESSING
042320  012767  000001  135244         MOV     #BIT0,SR0         ;TURN ON MEMORY MANAGEMENT
042326  105067  135256                 CLRB    UIPDR4            ;MAP USER SPACE NON-RESIDENT WHILE
042332  105067  135254                 CLRB    UIPDR5            ;        TESTING KERNEL SPACE
042336  010567  135306                 MOV     R5,UIPAR4         ;MAP USER PAR'S OPPOSITE OF KIPAR'S
042342  010467  135304                 MOV     R4,UIPAR5
042346  016767  135424  175004  4$:    MOV     PSW,$TMPO         ;SAVE PSW IN CASE OF ERROR
042354  012700  100100                 MOV     #100100,R0        ;PUT VIRTUAL ADDR. THAT USES PAR4 IN R0
042360  012701  120000                 MOV     #120000,R1        ;PUT VIRTUAL ADDR. THAT USES PAR5 IN R1
```

```
042364  010010                  5$:     MOV     R0,(R0)         ;WRITE TO TEST LOC. USING PAR4
042366  011102                          MOV     (R1),R2         ;READ THE SAME LOC., BUT USING PAR5
042370  020002                          CMP     R0,R2           ;DID WE READ WHAT WE WROTE?
042372  001401                          BEQ     6$
042374  104000                          EMT                     ;READING LOC. USING PAR5 AND A VIRT.
                                                                ;ADDR. DID NOT FIND DATA WRITTEN WHEN USING
                                                                ;PAR4 AND VIRT. ADDRESS.
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 5$'' = 000765
042376  062700  000100          6$:     ADD     #100,R0         ;CHANGE VIRTUAL ADDRS. TO POINT TO NEXT BLOCK
042402  062701  000100                  ADD     #100,R1
042406  020127  127700                  CMP     R1,#127700      ;WERE BLOCKS FROM 60000-676000 ALL TRIED?
042412  001364                          BNE     5$              ;BRANCH IF NO
042414  032767  140000  135354          BIT     #140000,PSW     ;HAVE WE DONE TEST IN USER MODE YET?
042422  001026                          BNE     7$              ;BRANCH IF YES
042424  010467  135220                  MOV     R4,UIPAR4       ;LOAD USER PAR4
042430  010567  135216                  MOV     R5,UIPAR5       ;LOAD USER PAR5
042434  112767  000006  135146          MOVB    #6,UIPDR4       ;MAP USER SPACE R/W TO TEST IT
042442  112767  000006  135142          MOVB    #6,UIPDR5
042450  105067  127634                  CLRB    KIPDR4          ;MAP  KERNEL SPACE NON-RESIDENT WHILE
042454  105067  127632                  CLRB    KIPDR5          ;       TESTING USER SPACE
042460  010567  127664                  MOV     R5,KIPAR4       ;MAP KERNEL PAR'S OPPOSITE UIPAR'S
042464  010467  127662                  MOV     R4,KIPAR5
042470  012767  140000  135300          MOV     #140000,PSW     ;GO TO USER MODE
042476  000723                          BR      4$              ;GO BACK AND READ/WRITE IN USER MODE
042500  005067  135272          7$:     CLR     PSW             ;GO BACK TO KERNEL MODE BEFORE LEAVING
042504  012767  077406  127576          MOV     #77406,KIPDR4   ;REMAP KERNEL PAGES READ/WRITE
042512  012767  077406  127572          MOV     #77406,KIPDR5
042520  010567  127624                  MOV     R5,KIPAR4       ;MAP KERNEL  AND USER PAR'S 4 & 5
042524  010567  127622                  MOV     R5,KIPAR5       ;  BACK TO 12-16K
042530  010567  135114                  MOV     R5,UIPAR4
042534  010567  135112                  MOV     R5,UIPAR5
042540  012767  021356  135502          MOV     #T0250,MMVEC    ;RESTORE ADDR. OF NORMAL M.M. TRAP ROUTINE
042546  000404                          BR      TS376           ;GET TO NEXT TEST
042550  042767  160000  135014  8$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
042556  104000                          EMT                     ;M.M. TRAP WHILE IN RELOCATE MODE -
                                                                ;REFERENCED WRONG SET OF PDR'S
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;A 'NOP'' = 000240

                                ;*******************************************************************************
                                ;TEST 376          W-BIT LOGIC TEST, KERNEL PDR'S
                                ;*******************************************************************************
042560                          TS376:
042560                          1$:
042560  004767  005262                  JSR     PC,TOFF         ;TURN T-BIT TRAPPING OFF FOR THIS TEST
042564  012702  000004                  MOV     #4,R2           ;SET LOOP COUNTER TO 4
042570  012700  172346                  MOV     #KIPAR3,R0      ;LOAD ADDRESS OF PAR3 INTO R0
042574  012701  001400                  MOV     #1400,R1        ;LOAD ''24-28K'' PAR VALUE INTO R1
042600  010120                  2$:     MOV     R1,(R0)+        ;MAP PARS 3-6 TO 12-16K
042602  077202                          SOB     R2,2$           ;LOOP TIL ALL 4 OF THEM LOADED
042604  012705  172300                  MOV     #KIPDR0,R5      ;LOAD ADDRESS OF FIRST PDR TO BE TESTED IN R5
042610  012704  000010                  MOV     #10,R4          ;SET LOOP COUNTER TO 8
042614  012703  017776                  MOV     #17776,R3       ;INITIALIZE VIRTUAL ADDRESS TO BE IN R3
```

```
042620  012700  172300          3$:     MOV     #KIPDRO,RO          ;LOAD ADDR. OF FIRST PDR TO BE SETUP IN RO
042624  012702  000010                  MOV     #10,R2              ;SET LOOP COUNTER TO 8
042630  012701  077406                  MOV     #77406,R1           ;PUT "W-BIT OFF DATA" INTO R1
042634  010120                  4$:     MOV     R1,(RO)+            ;CLEAR ALL W-BITS BY WRITING TO ALL PDRS
042636  077202                          SOB     R2,4$               ;LOOP UNTIL ALL OF THEM SETUP
042640  011313                          MOV     (R3),(R3)           ;DO "DATO" TO VIRTUAL ADDR.-SETTING A W-BIT
042642  031527  000100                  BIT     (R5),#WBIT          ;DID THAT CAUSE W-BIT TO BE SET?
042646  001001                          BNE     5$
042650  104000                          EMT                         ;W-BIT DID NOT GET SET IN PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 3$" = 000763
042652  012702  000010          5$:     MOV     #10,R2              ;SET LOOP COUNTER TO 8
042656  012700  172300                  MOV     #KIPDRO,RO          ;LOAD ADDR. OF FIRST PDR TO BE CHECKED IN RO
042662  031027  000100          6$:     BIT     (RO),#WBIT          ;DID W-BIT IN OTHER PDRS REMAIN CLEAR?
042666  001403                          BEQ     7$                  ;BRANCH IF YES
042670  020500                          CMP     R5,RO               ;IF W-BIT SET, THEN WAS IT PDR UNDER TEST?
042672  001401                          BEQ     7$
042674  104000                          EMT                         ;W-BIT GOT SET IN MORE THAN ONE PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 3$" = 000750
042676  062700  000002          7$:     ADD     #2,RO               ;POINT RO TO NEXT PDR TO BE CHECKED
042702  077211                          SOB     R2,6$               ;LOOP UNTIL ALL 8 CHECKED FOR CLEAR W-BIT
042704  010115                          MOV     R1,(R5)             ;WRITE TO THE PDR TESTED TO CLEAR W-BIT
042706  031527  000100                  BIT     (R5),#WBIT          ;DID WRITING PDR CLEAR THE W-BIT?
042712  001401                          BEQ     8$
042714  104000                          EMT                         ;W-BIT DID NOT CLEAR BY WRITING THE PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 3$" = 000740
042716  062705  000002          8$:     ADD     #2,R5               ;POINT R5 TO THE NEXT PDR TO BE TESTED
042722  062703  020000                  ADD     #20000,R3           ;CHANGE VIRT. ADDR TO REF. NEXT PDR
042726  077444                          SOB     R4,3$               ;LOOP BACK TO 3$ UNTIL ALL 8 PDR'S TESTED
042730  004767  005146                  JSR     PC,TON              ;TURN T-BIT BACK ON FOR NEXT TEST

                                ;*******************************************************************************
                                ;TEST 377       W-BIT LOGIC TEST, USER PDR'S
                                ;*******************************************************************************
042734                          TS377:
042734  012767  140000  135034  1$:     MOV     #140000,PSW         ;GO TO USER MODE FOR THIS TEST
042742  004767  005100                  JSR     PC,TOFF             ;TURN T-BIT TRAPPING OFF FOR THIS TEST
042746  012702  000004                  MOV     #4,R2               ;SET LOOP COUNTER TO 4
042752  012700  177646                  MOV     #UIPAR3,RO          ;LOAD ADDRESS OF PAR3 INTO RO
042756  012701  001400                  MOV     #1400,R1            ;LOAD "24-28K" PAR VALUE INTO R1
042762  010120                  2$:     MOV     R1,(RO)+            ;MAP PARS 3-6 TO 12-16K
042764  077202                          SOB     R2,2$               ;LOOP TIL ALL 4 OF THEM LOADED
042766  012705  177600                  MOV     #UIPDRO,R5          ;LOAD ADDRESS OF FIRST PDR TO BE TESTED IN R5
042772  012704  000010                  MOV     #10,R4              ;SET LOOP COUNTER TO 8
042776  012703  017776                  MOV     #17776,R3           ;INITIALIZE VIRTUAL ADDRESS TO BE IN R3
043002  012700  177600          3$:     MOV     #UIPDRO,RO          ;LOAD ADDR. OF FIRST PDR TO BE SETUP IN RO
043006  012702  000010                  MOV     #10,R2              ;SET LOOP COUNTER TO 8
043012  012701  077406                  MOV     #77406,R1           ;PUT "W-BIT OFF DATA" INTO R1
043016  010120                  4$:     MOV     R1,(RO)+            ;CLEAR ALL W-BITS BY WRITING TO ALL PDRS
043020  077202                          SOB     R2,4$               ;LOOP UNTIL ALL OF THEM SETUP
043022  011313                          MOV     (R3),(R3)           ;DO "DATO" TO VIRTUAL ADDR.-SETTING A W-BIT
```

```
043024  031527  000100                  BIT     (R5),#WBIT              ;DID THAT CAUSE W-BIT TO BE SET?
043030  001001                          BNE     5$
043032  104000                          EMT                             ;W-BIT DID NOT GET SET IN PDR
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 3$" = 000763
043034  012702  000010          5$:     MOV     #10,R2                  ;SET LOOP COUNTER TO 8
043040  012700  177600                  MOV     #UIPDR0,R0              ;LOAD ADDR. OF FIRST PDR TO BE CHECKED IN R0
043044  031027  000100          6$:     BIT     (R0),#WBIT              ;DID W-BIT IN OTHER PDRS REMAIN CLEAR?
043050  001403                          BEQ     7$                      ;BRANCH IF YES
043052  020500                          CMP     R5,R0                   ;IF W-BIT SET, THEN WAS IT PDR UNDER TEST?
043054  001401                          BEQ     7$
043056  104000                          EMT                             ;W-BIT GOT SET IN MORE THAN ONE PDR
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 3$" = 000750
043060  062700  000002          7$:     ADD     #2,R0                   ;POINT R0 TO NEXT PDR TO BE CHECKED
043064  077211                          SOB     R2,6$                   ;LOOP UNTIL ALL 8 CHECKED FOR CLEAR W-BIT
043066  010115                          MOV     R1,(R5)                 ;WRITE TO THE PDR TESTED TO CLEAR W-BIT
043070  031527  000100                  BIT     (R5),#WBIT              ;DID WRITING PDR CLEAR THE W-BIT?
043074  001401                          BEQ     8$
043076  104000                          EMT                             ;W-BIT DID NOT CLEAR BY WRITING THE PDR
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 3$" = 000740
043100  062705  000002          8$:     ADD     #2,R5                   ;POINT R5 TO THE NEXT PDR TO BE TESTED
043104  062703  020000                  ADD     #20000,R3               ;CHANGE VIRT. ADDR TO REF. NEXT PDR
043110  077444                          SOB     R4,3$                   ;LOOP BACK TO 3$ UNTIL ALL 8 PDR'S TESTED
043112  004767  004764                  JSR     PC,TON                  ;TURN T-BIT BACK ON FOR NEXT TEST
043116  005067  134654                  CLR     PSW                     ;BACK TO KERNEL MODE BEFORE LEAVING

                                ;*****************************************************************************
                                ;TEST 400       TEST "W-BIT" SPECIAL CASES
                                ;*****************************************************************************
043122                          TS400:

043122  004767  004720          1$:     JSR     PC,TOFF                 ;TURN OFF T-BIT TRAPPING FOR THIS TEST
043126  012701  077406                  MOV     #77406,R1               ;PUT "W-BIT OFF" VALUE FOR PDR IN R1
043132  010167  127160          2$:     MOV     R1,KIPDR7               ;LOAD KERNEL PDR 7 TO CLEAR W-BIT
043136  016700  134430                  MOV     SR0,R0                  ;READ PRESENT CONTENTS OF STATUS REG. 0
043142  010067  134424                  MOV     R0,SR0                  ;WRITE PRESENT CONTENTS OF SR0 BACK TO ITSELF
043146  016702  127144                  MOV     KIPDR7,R2               ;READ CONTENTS OF KIPDR7 INTO R2
043152  020102                          CMP     R1,R2                   ;WAS W-BIT LEFT CLEARED?
043154  001401                          BEQ     3$
043156  104000                          EMT                             ;W-BIT IN KIPDR7 SET WHEN SR0 WAS WRITTEN TO
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 2$" = 000765
043160  010167  127130          3$:     MOV     R1,KIPDR6               ;LOAD KERNEL PDR6 WITH 77406 TO CLEAR W-BIT
043164  012767  043176  134612          MOV     #4$,ERRVEC              ;SET UP LOC. 4 TO 4$ FOR ODD ADDR. ABORT
043172  005037  140000                  CLR     @#140000                ;CAUSE TIMEOUT ABORT THRU LOC. 4
043176  012706  001000          4$:     MOV     #KERSTK,KSP             ;RESTORE THE STACK POINTER
043202  016702  127106                  MOV     KIPDR6,R2               ;READ KIPDR6 INTO R2
043206  052701  000100                  BIS     #100,R1                 ;R1-77506
043212  020102                          CMP     R1,R2                   ;WAS W-BIT SET?
043214  001401                          BEQ     5$
```

```
043216  104000                             EMT                       ;W-BIT WAS NOT SET DURING A TIMEOUT ABORT
                                                                     ;FOR TIGHTER SCOPE LOOP
                                                                     ;REPLACE ERROR CALL WITH
                                                                     ;'BR 3$'' = 000757
043220  010167  127070             5$:     MOV     R1,KIPDR6         ;RESTORE KIPDR6 TO 77406
043224  012767  001400  127122             MOV     #1400,KIPAR6      ;RESTORE KIPAR6 TO 1400
043232  012767  021336  134544             MOV     #T04,ERRVEC       ;RESTORE NORMAL CPU TRAP ROUTINE TO LOC.4
043240  004767  004636                     JSR     PC,TON            ;TURN T-BIT TRAPPING BACK ON

                                   ;*************************************************************************
                                   ;*
                                   ;*      THE NEXT THREE (3) TESTS CAUSE MEMORY MANAGEMENT ERRORS
                                   ;*      TO CHECK THE ABILITY OF STATUS REGISTER 0 TO RECORD KT
                                   ;*      ERRORS AND THE ABILITY OF STATUS REGISTER 2 TO LOCK UP THE
                                   ;*      VIRTUAL ADDR. OF THE INSTRUCTION THAT CAUSED THE ERROR.
                                   ;*      THE BITS OF SR2 ARE CHECKED AND BITS <15:13>, <6:5>, AND <3:0>
                                   ;*      ARE CHECKED IN SR0.  SO THE SR0 AND SR2 LOGIC AND THE
                                   ;*      KT ERROR LOGIC ARE CHECKED.
                                   ;*
                                   ;*************************************************************************

                                   ;*************************************************************************
                                   ;TEST 401        NON-RESIDENT ABORT TEST (ACF=0&4)
                                   ;*************************************************************************
043244                             TS401:

043244  012700  001400             1$:     MOV     #1400,R0          ;LOAD DATA FOR PAR'S INTO R0
043250  010067  127072                     MOV     R0,KIPAR3         ;MAP KERNEL PAR'S 3&4 TO 24-28K
043254  010067  127070                     MOV     R0,KIPAR4
043260  010067  134362                     MOV     R0,UIPAR3         ;MAP USER PAR'S 3&4 TO 24-28K
043264  010067  134360                     MOV     R0,UIPAR4
043270  012767  077406  127010             MOV     #77406,KIPDR3     ;MAP KERNEL PDR 3  128 BLKS, READ-WRITE
043276  012767  077406  134302             MOV     #77406,UIPDR3     ;MAP USER PDR 3  128 BLKS, READ-WRITE
043304  012700  060000                     MOV     #60000,R0         ;LOAD VIRTUAL ADDR. TO REFERENCE PDR3 INTO R0
043310  012701  100000                     MOV     #100000,R1        ;LOAD VIRTUAL ADDR. TO REFERENCE PDR4 INTO R1
043314  012703  100011                     MOV     #100011,R3        ;LOAD R3 WITH WHAT SR0 SHOULD READ - N.R., KERNEL, PG.4
043320  012702  077400                     MOV     #77400,R2         ;LOAD ACF=0 (NON-RESIDENT) PDR VALUE IN R2
043324  012767  043360  134716     2$:     MOV     #5$,MMVEC         ;POINT MEM. MGMT. TRAP VECTOR TO 5$ BELOW
043332  010267  126752                     MOV     R2,KIPDR4         ;LOAD ACF TEST VALUE INTO KIPDR4
043336  010267  134246                     MOV     R2,UIPDR4         ;LOAD ACF TEST VALUE INTO UIPDR4
043342  005010                     3$:     CLR     (R0)              ;CLEAR PHYS. LOC. 140000 USING PDR3
043344  016767  134426  174006             MOV     PSW,$TMP0         ;SAVE PSW IN CASE OF ERROR
043352  005211                     4$:     INC     (R1)              ;TRY TO REF. IT USING PDR4 - SHOULD TRAP TO 5$
043354  001462                             BEQ     TS402
043356  104000                             EMT                       ;MEM. MGMT. ABORT DID NOT OCCUR
                                                                     ;FOR TIGHTER SCOPE LOOP
                                                                     ;REPLACE ERROR CALL WITH
                                                                     ;'BR 3$'' = 000772
043360  062706  000004             5$:     ADD     #4,SP             ;RESTORE STACK POINTER
043364  005710                             TST     (R0)              ;DID INSTRUCTION GET ABORTED & NOT EXECUTE
043366  001401                             BEQ     6$
043370  104000                             EMT                       ;INSTRUCTION WAS NOT ABORTED, LOC. GOT CHANGED
                                                                     ;FOR TIGHTER SCOPE LOOP
                                                                     ;REPLACE ERROR CALL WITH
                                                                     ;'BR 3$'' = 000764
043372  016767  134174  173752     6$:     MOV     SR0,WASSR0        ;READ STATUS REGISTER 0
```

```
043400  016767  134172  173746          MOV     SR2,WASSR2          ;READ STATUS REGISTER 2
043406  020367  173740                  CMP     R3,WASSR0           ;DID SR0 REPORT NON-RESIDENT ERROR CORRECTLY?
043412  001401                          BEQ     7$
043414  104000                          EMT                         ;SR0 DID NOT REPORT NON-RES. ERROR CORRECTLY
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 3$" = 000752
043416  012704  043352          7$:     MOV     #4$,R4              ;LOAD R4 WITH WHAT SR2 SHOULD READ
043422  020467  173726                  CMP     R4,WASSR2           ;DID SR2 LOCKUP RIGHT VIRTUAL ADDR. (=4$)?
043426  001401                          BEQ     8$
043430  104000                          EMT                         ;SR2 DID NOT LOCK VIRTUAL ADDR. OF NON-RES. ERROR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 3$" = 000744
043432  042767  160000  134132  8$:     BIC     #160000,SR0         ;CLEAR THE ERROR BITS IN SR0
043440  032767  140000  173712          BIT     #140000,$TMP0       ;HAS ACF=0&4 BEEN TESTED IN USER YET
043446  001006                          BNE     9$                  ;BRANCH IF YES
043450  012703  100151                  MOV     #100151,R3          ;LOAD R3 WITH WHAT SR0 SHOULD READ - N.R., USER, PG.4
043454  012767  140000  134314          MOV     #140000,PSW         ;GO TO USER MODE
043462  000720                          BR      2$                  ;REPEAT TEST IN USER MODE
043464  022702  077404          9$:     CMP     #77404,R2           ;HAS ACF=4 BEEN TESTED YET?
043470  001407                          BEQ     10$                 ;BRANCH IF YES
043472  012702  077404                  MOV     #77404,R2           ;THEN LOAD ACF=4 (NON-RES) PDR VALUE IN R2
043476  012703  100011                  MOV     #100011,R3          ;LOAD R3 WITH WHAT SR0 SHOULD READ-N.R.,KERNEL,PG. 4
043502  005067  134270                  CLR     PSW                 ;GO BACK TO KERNEL MODE
043506  000706                          BR      2$                  ;GO BACK & TEST ACF=4 IN SAME MODE
043510  005067  134262          10$:    CLR     PSW                 ;GO BACK TO KERNEL MODE BEFORE LEAVING
043514  012767  021356  134526          MOV     #T0250,MMVEC        ;RESTORE ADDRESS OF NORMAL MEMORY
                                                                    ;MANAGEMENT ERROR ROUTINE TO MMVEC

                                ;************************************************************************
                                ;TEST 402       READ-ONLY ABORT TEST (ACF=2)
                                ;************************************************************************
043522                          TS402:
043522                          1$:                                 ;KERNEL & USER PAR'S 3 & 4 AND PDR 3
                                                                    ;ARE SETUP FROM LAST TEST
043522  012700  060000                  MOV     #60000,R0           ;LOAD VIRTUAL ADDR. TO REFERENCE PDR3 INTO R0
043526  012701  100000                  MOV     #100000,R1          ;LOAD VIRTUAL ADDR. TO REFERENCE PDR4 INTO R1
043532  012703  020011                  MOV     #20011,R3           ;LOAD R3 WITH WHAT SR0 SHOULD READ - R/O, KERNEL, PG.4
043536  012702  077402                  MOV     #77402,R2           ;LOAD ACF=2 (READ-ONLY) PDR VALUE IN R2
043542  012767  043574  134500  2$:     MOV     #5$,MMVEC           ;POINT MEM. MGMT. TRAP VECTOR TO 5$ BELOW
043550  010267  126534                  MOV     R2,KIPDR4           ;LOAD ACF=2 INTO KIPDR4
043554  010267  134030                  MOV     R2,UIPDR4           ;LOAD ACF=2 INTO UIPDR4
043560  005010                  3$:     CLR     (R0)                ;CLEAR PHYS. LOC. 140000 USING PDR3
043562  016767  134210  173570          MOV     PSW,$TMP0           ;SAVE PSW IN CASE OF ERROR
043570  005211                  4$:     INC     (R1)                ;TRY TO WRITE USING PDR4 - SHOULD TRAP TO 5$
043572  104000                          EMT                         ;MEM. MGMT. ABORT DID NOT OCCUR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;'BR 3$" = 000772
043574  062706  000004          5$:     ADD     #4,SP               ;RESTORE STACK POINTER
043600  005710                          TST     (R0)                ;DID INSTRUCTION GET ABORTED & NOT EXECUTE
043602  001401                          BEQ     6$
043604  104000                          EMT                         ;INSTRUCTION WAS NOT ABORTED, LOC. GOT CHANGED
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
```

```
                                                                        ;'BR 3$'' = 000764
043606  016767  133760  173536  6$:     MOV     SR0,WASSR0              ;READ STATUS REG. 0
043614  016767  133756  173532          MOV     SR2,WASSR2              ;READ STATUS REG. 2
043622  020367  173524                  CMP     R3,WASSR0               ;DID SR0 REPORT READ-ONLY ERROR CORRECTLY?
043626  001401                          BEQ     7$
043630  104000                          EMT                             ;SR0 DID NOT REPORT R/O ERROR CORRECTLY
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;'BR 3$'' = 000752
043632  012704  043570          7$:     MOV     #4$,R4                  ;LOAD R4 WITH WHAT SR2 SHOULD READ
043636  020467  173512                  CMP     R4,WASSR2               ;DID SR2 LOCKUP RIGHT VIRTUAL ADDR. (=4$)?
043642  001401                          BEQ     8$
043644  104000                          EMT                             ;SR2 DID NOT LOCKUP VIRTUAL ADDR. OF R/O ERROR
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;'BR 3$'' = 000744
043646  042767  160000  133716  8$:     BIC     #160000,SR0             ;CLEAR THE ERROR BITS IN SR0
043654  032767  140000  173476          BIT     #140000,$TMP0           ;HAS ACF=2 BEEN TESTED IN USER MODE?
043662  001006                          BNE     9$                      ;BRANCH IF YES
043664  012703  020151                  MOV     #20151,R3               ;LOAD R3 WITH WHAT SR0 SHOULD READ-R/O, USER, PG.4
043670  012767  140000  134100          MOV     #140000,PSW             ;GO TO USER MODE
043676  000721                          BR      2$                      ;REPEAT TEST IN USER MODE
043700  005067  134072          9$:     CLR     PSW                     ;GO BACK TO KERNEL MODE BEFORE LEAVING
043704  012767  021356  134336          MOV     #T0250,MMVEC            ;RESTGRE ADDRESS OF NORMAL MEMORY
                                                                        ;MANAGEMENT ERROR ROUTINE TO MMVEC.




                                ;NOTE:  MACRO MSG31A WAS DELETED AS IT DIDN'T APPLY TO F11.









                                ;*************************************************************************
                                ;*
                                ;*      THE NEXT TWO (2) TESTS WILL BE CHECKING THE PAGE LENGTH
                                ;*      COMPARATORS AND SOME MORE OF THE KT ERROR DETECTION
                                ;*      AND STATUS LOGIC.  THE PAGE LENGTH FIELD (PLF) IN KERNEL
                                ;*      PDR 4 IS VARIED AND FOR EVERY PLF, THREE (3) VIRTUAL
                                ;*      ADDRESSES ARE READ.  WHILE USING BOTH UPWARD & DOWNWARD PAGE
                                ;*      EXPANSION, ONE OF THOSE THREE VIRTUAL ADDRESSES WILL CAUSE A
                                ;*      ''PAGE LENGTH ABORT'' WHILE THE OTHER TWO WON'T.
                                ;*
                                ;*      STATUS REGISTER 0 & 2 ARE CHECKED WHEN THE PAGE LENGTH
                                ;*      ABORT DOES OCCUR TO SEE THAT THE ABORT IS REPORTED AND THAT
                                ;*      THE VIRTUAL ADDRESS OF THE INSTRUCTION THAT CAUSED THE ABORT
                                ;*      IS LOCKED UP.
                                ;*
                                ;*************************************************************************
```

```
                                    ;*************************************************************************
                                    ;TEST 403       PAGE LENGTH FAULTS-UPWARD EXPANSION
                                    ;*************************************************************************
043712                              TS403:
043712  012767  077406  126366      1$:     MOV     #77406,KIPDR3   ;MAKE SURE PDR3 IS DESCRIBED AS R/W
043720  012767  077406  126364              MOV     #77406,KIPDR5   ;MAKE SURE PDR5 IS DESCRIBED AS R/W
043726  012700  044126                      MOV     #DALTB1,R0      ;DAL TABLE FOR VIRTUAL ADDR'S. TO SELECT PDR4.
043732  012704  044144                      MOV     #PDRTB1,R4      ;PDR TABLE FOR PDR4 (COINCIDES WITH DAL TABLE).
043736  012701  000006                      MOV     #6,R1           ;SET UP LOOP COUNTER.
043742  012767  044104  134300              MOV     #9$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR UNEXPECTED ABORTS
043750  012706  001000                      MOV     #KERSTK,KSP     ;MAKE SURE STACK POINTER IS ALL SET UP

                                    ;TEST NON-ABORT CASES (VBA < OR = PLF)
043754  012467  126330              2$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGTH VALUE
043760  005730                              TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA < OR = PLF)
                                                                    ;NO ABORT SHOULD OCCUR!!!
043762  077104                              SOB     R1,2$           ;DONE?...NO- TEST NEXT COMBINATION OF DAL & PDR.

                                    ;TEST ABORT CASES (VBA > PLF)
043764  012701  000005              3$:     MOV     #5,R1           ;SET UP LOOP COUNTER.
043770  012700  044162                      MOV     #DALTB2,R0      ;DAL TABLE
043774  012704  044176                      MOV     #PDRTB2,R4      ;PDR TABLE
044000  012767  044020  134242              MOV     #6$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR EXPECTED ABORT

044006  012467  126276              4$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGHT VALUE
044012  005730                      5$:     TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA > PLF - ABORT TO 6$)
044014  001476                              BEQ     TS404
044016  104000                              EMT                     ;EXPECTED PAGE LENGTH ABORT DID NOT OCCUR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 5$" = 000776
044020  012706  001000              6$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER FOLLOWING ABORT
044024  016767  133542  173320              MOV     SR0,WASSR0      ;READ M.M. STATUS REG. 0
044032  016767  133540  173314              MOV     SR2,WASSR2      ;READ M.M. STATUS REG. 2
044040  012702  040011                      MOV     #40011,R2       ;PUT EXPECTED SR0 CONTENTS IN R2
044044  020267  173302                      CMP     R2,WASSR0       ;DID SR0 REPORT PG. LENGTH ABORT, PAGE 4, KERNEL?
044050  001401                              BEQ     7$
044052  104000                              EMT                     ;SR0 DID NOT REPORT PG. LENGTH ABORT CORRECTLY
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 5$" = 000757
044054  012703  044012              7$:     MOV     #5$,R3          ;PUT EXPECTED SR2 CONTENTS IN R3
044060  020367  173270                      CMP     R3,WASSR2       ;DID SR2 LOCKUP VIRT. ADDR. OF ABORTED INSTRUCTION?
044064  001401                              BEQ     8$
044066  104000                              EMT                     ;SR2 DID NOT LOCKUP VIRT. ADDR. OF ABORT CORRECTLY
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 5$" = 000751
044070  042767  160000  133474      8$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
044076  077135                              SOB     R1,4$           ;DONE?..NO - GET NEXT DAL & PDR PAIR
044100  000167  000010                      JMP     10$             ;YES...
044104  042767  160000  133460      9$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
044112  104000                              EMT                     ;GOT PG. LENGTH ABORT BEFORE IT WAS EXPECTED
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;A "NOP" = 240
```

```
 044114  012767  021356  134126  10$:    MOV     #T0250,MMVEC     ;RESTORE NORMAL M.M. TRAP HANDLER
                                                                  ;ADDRESS TO M.M. TRAP VECTOR
 044122  000167  000064                  JMP     TS404            ;GET TO NEXT TEST

                                 ;DAL TABLE FOR UPWARD EXPANSION (NON-ABORT CASES)
 044126  100000                  DALTB1: 100000
 044130  106100                          106100
 044132  102300                          102300
 044134  102500                          102500
 044136  113700                          113700
 044140  104600                          104600
 044142  117700                          117700

                                 ;PDR TABLE FOR KPDR4 (NON-ABORT CASES)
 044144  000006                  PDRTB1: 000006
 044146  052006                          052006
 044150  045006                          045006
 044152  052006                          052006
 044154  074406                          074406
 044156  025006                          025006
 044160  077406                          077406

                                 ;DAL TABLE (ABORT CASES)
 044162  100100                  DALTB2: 100100
 044164  110100                          110100
 044166  116600                          116600
 044170  112700                          112700
 044172  117000                          117000
 044174  117700                          117700

                                 ;PDR TABLE (ABORT CASES)
 044176  000006                  PDRTB2: 000006
 044200  030406                          030406
 044202  046406                          046406
 044204  042006                          042006
 044206  073406                          073406
 044210  077006                          077006



                                 ;*******************************************************************************
                                 ;TEST 404         PAGE LENGTH FAULTS-DOWNWARD EXPANSION
                                 ;*******************************************************************************
 044212                          TS404:
 044212  012700  044412          1$:     MOV     #DALTB3,R0       ;DAL TABLE FOR VIRTUAL ADDR'S. TO SELECT PDR4.
 044216  012704  044430                  MOV     #PDRTB3,R4       ;PDR TABLE FOR PDR4 (COINCIDES WITH DAL TABLE).
 044222  012701  000006                  MOV     #6,R1            ;SET UP LOOP COUNTER.
 044226  012767  044370  134014          MOV     #9$,MMVEC        ;SETUP M.M. TRAP VECTOR FOR UNEXPECTED ABORTS
 044234  012706  001000                  MOV     #KERSTK,KSP      ;MAKE SURE STACK POINTER IS ALL SET UP

                                 ;TEST NON-ABORT CASES (VBA > OR = PLF)
 044240  012467  126044          2$:     MOV     (R4)+,KIPDR4     ;LOAD KIPDR4 WITH PAGE LENGTH VALUE
 044244  005730                          TST     @(R0)+           ;ACCESS VIRTUAL ADDR. (VBA > OR = PLF)
                                                                  ;NO ABORT SHOULD OCCUR!!!
 044246  077104                          SOB     R1,2$            ;DONE?...NO- TEST NEXT COMBINATION OF DAL & PDR.
```

```
                                        ;TEST ABORT CASES (VBA < PLF)
044250  012701  000005          3$:     MOV     #5,R1           ;SET UP LOOP COUNTER.
044254  012700  044446                  MOV     #DALTB4,R0      ;DAL TABLE
044260  012704  044462                  MOV     #PDRTB4,R4      ;PDR TABLE
044264  012767  044304  133756          MOV     #6$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR EXPECTED ABORT

044272  012467  126012          4$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGHT VALUE
044276  005730                  5$:     TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA < PLF - ABORT TO 6$)
044300  001476                          BEQ     TS405
044302  104000                          EMT                     ;EXPECTED PAGE LENGTH ABORT DID NOT OCCUR
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 5$' = 000776
044304  012706  001000          6$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER FOLLOWING ABORT
044310  016767  133256  173034          MOV     SR0,WASSR0      ;READ M.M. STATUS REG. 0
044316  016767  133254  173030          MOV     SR2,WASSR2      ;READ M.M. STATUS REG. 2
044324  012702  040011                  MOV     #40011,R2       ;PUT EXPECTED SR0 CONTENTS IN R2
044330  020267  173016                  CMP     R2,WASSR0       ;DID SR0 REPORT PG. LENGTH ABORT, PAGE 4, KERNEL?
044334  001401                          BEQ     7$
044336  104000                          EMT                     ;SR0 DID NOT REPORT PG. LENGTH ABORT CORRECTLY
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 5$' = 000757
044340  012703  044276          7$:     MOV     #5$,R3          ;PUT EXPECTED SR2 CONTENTS IN R3
044344  020367  173004                  CMP     R3,WASSR2       ;DID SR2 LOCKUP VIRT. ADDR. OF ABORTED INSTRUCTION?
044350  001401                          BEQ     8$
044352  104000                          EMT                     ;SR2 DID NOT LOCKUP VIRT. ADDR. OF ABORT CORRECTLY
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 5$' = 000751
044354  042767  160000  133210  8$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
044362  077135                          SOB     R1,4$           ;DONE?..NO - GET NEXT DAL & PDR PAIR
044364  000167  000010                  JMP     10$             ;YES...
044370  042767  160000  133174  9$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
044376  104000                          EMT                     ;GOT PG. LENGTH ABORT BEFORE IT WAS EXPECTED
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;A 'NOP' = 000240

044400  012767  021356  133642  10$:    MOV     #T0250,MMVEC    ;RESTORE NORMAL M.M. TRAP HANDLER
                                                                ;ADDRESS TO M.M. TRAP VECTOR
044406  000167  000064                  JMP     TS405           ;GET TO NEXT TEST

                                        ;DAL TABLE FOR DOWNWARD EXPANSION (NON-ABORT CASES)
044412  117700                          DALTB3: 117700
044414  111600                                  111600
044416  115400                                  115400
044420  115200                                  115200
044422  104000                                  104000
044424  113100                                  113100
044426  100000                                  100000

                                        ;PDR TABLE (NON-ABORT CASES)
044430  077416                          PDRTB3: 77416
044432  025416                                  25416
044434  032416                                  32416
```

```
044436  025416                                  25416
044440  003016                                  03016
044442  052416                                  52416
044444  000016                                  00016

                                        ;DAL TABLE (ABORT CASES)
044446  117600                          DALTB4: 117600
044450  107600                                  107600
044452  101100                                  101100
044454  105000                                  105000
044456  100700                                  100700
044460  100000                                  100000

                                        ;PDR TABLE (ABORT CASES)
044462  077416                          PDRTB4: 77416
044464  047016                                  47016
044466  031016                                  31016
044470  035416                                  35416
044472  004016                                  04016
044474  000416                                  00416



                                        ;*******************************************************************************
                                        ;TEST 405           SR2 BIT TEST
                                        ;*******************************************************************************
044476                                  TS405:
044476  012767  001400  125642          1$:     MOV     #1400,KIPAR3    ;BE SURE PAR3 IS MAPPED TO 24-28K
044504  012767  001400  125636                  MOV     #1400,KIPAR4    ;BE SURE PAR4 IS MAPPED TO 24-28K
044512  012767  077406  125566                  MOV     #77406,KIPDR3   ;MAP PAGE 3  128 BLOCKS, R/W
044520  012767  077402  125562                  MOV     #77402,KIPDR4   ;MAP PAGE 4  128 BLOCKS, READ-ONLY
044526  012700  060002                          MOV     #60002,R0       ;LOAD R0 WITH VIRTUAL ADDR. WHICH USES PDR3
044532  012701  100002                          MOV     #100002,R1      ;LOAD R1 WITH VIRTUAL ADDR. WHICH USES PDR4
044536  012767  044564  133504                  MOV     #3$,MMVEC       ;SET M.M. TRAP VECTOR TO 3$
044544  012720  010727                  2$:     MOV     #010727,(R0)+   ;LOAD "MOV PC,(PC)+" INSTRUCTION AT ADDR.
044550  005020                                  CLR     (R0)+           ;          REACHED THRU PDR/PAR 4.
044552  012720  000137                          MOV     #000137,(R0)+   ;LOAD "JMP @#3$" INSTRUCTION AT VIRT. ADDR.
044556  012710  044564                          MOV     #3$,(R0)        ;          IN CASE R/O VIOL. DOES NOT ABORT
044562  010107                                  MOV     R1,PC           ;TRANSFER PROGRAM EXECUTION TO "PAGE 4 INSTRUCTIONS"
044564  012706  001000                  3$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER
044570  016767  133002  172556                  MOV     SR2,WASSR2      ;READ CONTENTS OF STATUS REG 2
044576  020167  172552                          CMP     R1,WASSR2       ;WAS ADDR. OF "RELOCATED - R/O ABORT" LOCKED UP?
044602  001401                                  BEQ     4$
044604  104000                                  EMT                     ;SR2 DID NOT LOCK UP VIRTUAL ADDR. OF R/O VIOL.
                                                                        ;FOR TIGHTER SCOPE LOOP
                                                                        ;REPLACE ERROR CALL WITH
                                                                        ;"BR 2$" = 000757
044606  042767  160000  132756          4$:     BIC     #160000,SR0     ;CLEAR THE ERROR BITS IN SR0
044614  060101                                  ADD     R1,R1           ;SETUP TO FORM NEXT VIRTUAL ADDRESS
044616  010100                                  MOV     R1,R0           ;SETUP R0 TO FORM NEXT VIRT. ADDR. TO LOAD
044620  052701  100000                          BIS     #100000,R1      ;FORM VIRTUAL ADDR. THAT SHOULD BE LOCKED UP NEXT
044624  052700  060000                          BIS     #60000,R0       ;POINT R0 TO NEXT VIRT. ADDR. TO LOAD
044630  020127  110000                          CMP     R1,#110000      ;HAVE ALL VBA'S 100000-110000 BEEN TESTED?
044634  101743                                  BLOS    2$              ;BRANCH IF NO

044636  012767  077406  125444                  MOV     #77406,KIPDR4   ;RESTORE PDR4 TO R/W ACCESS
```

```
044644  012767  021356  133376          MOV     #T0250,MMVEC        ;RESTORE ADDRESS OF NORMAL M.M.
                                                                    ;TRAP HANDLER TO M.M. VECTOR



                                ;*******************************************************************************
                                ;TEST 406               MORE CHECKS OF SR0 & SR2
                                ;*******************************************************************************
044652                          TS406:
044652  012767  001400  125472  1$:     MOV     #1400,KIPAR5        ;MAP KERNEL PAGE 5 TO 24-28K
044660  012767  000406  125422          MOV     #406,KIPDR4         ;SETUP PDR4 FOR PAGE LENGTH ABORT
044666  012767  077402  125416          MOV     #77402,KIPDR5       ;SETUP PDR5 FOR R/O ABORT
044674  016767  132676  172452  2$:     MOV     SR2,WASSR2          ;READ SR2 TO SEE IF ITS TRACKING
044702  012701  044674                  MOV     #2$,R1              ;PUT EXPECTED VIRTUAL PC IN R1
044706  020167  172442                  CMP     R1,WASSR2           ;DID SR2 CONTAIN VIRTUAL PC AT 2$?
044712  001401                          BEQ     4$
044714  104000                          EMT                         ;SR2 NOT TRACKING CORRECTLY
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 2$" = 000767
044716  016767  132654  172430  4$:     MOV     SR2,WASSR2          ;READ SR2 TO SEE IF ITS TRACKING
044724  012701  044716                  MOV     #4$,R1              ;PUT EXPECTED VIRTUAL PC IN R1
044730  020167  172420                  CMP     R1,WASSR2           ;DID SR2 CONTAIN VIRTUAL PC AT 4$
044734  001401                          BEQ     6$
044736  104000                          EMT                         ;SR2 NOT TRACKING CORRECTLY
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 4$" = 000767
044740  012767  044756  133302  6$:     MOV     #7$,MMVEC           ;PUT ADDRESS OF 7$ IN M.M. TRAP VECTOR
044746  005067  172410                  CLR     $TMP1               ;CLEAR ERROR INDICATOR
044752  005237  100500                  INC     @#100500            ;CAUSE PAGE LENGTH ABORT - TRAP TO 7$
044756  012706  001000          7$:     MOV     #KERSTK,KSP         ;RESTORE STACK POINTER AFTER ABORT
044762  016767  132604  172370          MOV     SR0,$TMP0           ;SAVE SR0'S INFORMATION ON PG. LGTH. ABORT
044770  016767  132602  172366          MOV     SR2,$TMP2           ;SAVE SR2'S INFORMATION ON PG. LGTH. ABORT
044776  012767  045010  133244          MOV     #8$,MMVEC           ;PUT ADDRESS OF 8$ IN M.M. TRAP VECTOR
045004  005237  120000                  INC     @#120000            ;CAUSE R/O ABORT - TRAP TO 8$
045010  012706  001000          8$:     MOV     #KERSTK,KSP         ;RESTORE STACK POINTER AFTER ABORT
045014  016767  132552  172330          MOV     SR0,WASSR0          ;READ SR0 FOLLWOING SECOND KT ABORT
045022  016767  132550  172324          MOV     SR2,WASSR2          ;READ SR2 FOLLOWING SECOND KT ABORT
045030  026767  172324  172314          CMP     $TMP0,WASSR0        ;IS SR0 STILL HOLDING INFO ON FIRST ABORT?
045036  001402                          BEQ     9$                  ;BRANCH IF YES
045040  005267  172316                  INC     $TMP1               ;SET ERROR INDICATOR
045044  026767  172314  172302  9$:     CMP     $TMP2,WASSR2        ;DOES SR2 STILL HOLD PC OF FIRST ABORT?
045052  001402                          BEQ     10$                 ;BRANCH IF YES
045054  005267  172302                  INC     $TMP1               ;SET ERROR INDICATOR
045060  005767  172276          10$:    TST     $TMP1               ;WERE SR0 OR SR2 CHANGED BY A SECOND ABORT?
045064  001401                          BEQ     11$
045066  104000                          EMT                         ;ONE OF STATUS REGS. CHANGED BY SECOND ABORT
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 6$" = 000726
045070  005067  172266          11$:    CLR     $TMP1               ;CLEAR ERROR INDICATOR
045074  000005                          RESET                       ;EXECUTE A RESET, APPLYING AN "INIT"
045076  005067  132470                  CLR     SR0
045102  016767  132464  172242          MOV     SR0,WASSR0          ;READ SR0
045110  005767  172236                  TST     WASSR0              ;WAS SR0 CLEARED BY THE RESET?
045114  001402                          BEQ     12$                 ;BRANCH IF YES
```

```
045116  005267  172240                    INC     $TMP1           ;SR0 NOT CLEARED BY A RESET
045122  016767  132450  172224    12$:    MOV     SR2,WASSR2      ;READ SR2
045130  022767  045122  172216            CMP     #12$,WASSR2     ;WAS SR2 UNLOCKED BY A RESET?
045136  001402                            BEQ     13$             ;BRANCH IF YES
045140  005267  172216                    INC     $TMP1           ;SR2 NOT UNLOCKED BY A RESET
045144  005767  172212            13$:    TST     $TMP1           ;WERE SR0 & SR2 BOTH "RESET" BY A RESET?
045150  001401                            BEQ     14$
045152  104000                            EMT                     ;SR0 OR SR2 NOT "RESET" BY A RESET
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 6$" = 000676
045154  012767  000001  132410    14$:    MOV     #1,SR0          ;TURN MEMORY MANAGEMENT BACK ON
045162  016767  132410  172164    15$:    MOV     SR2,WASSR2      ;READ SR2 TO SEE IF ITS TRACKING AGAIN
045170  012701  045162                    MOV     #15$,R1         ;PUT EXPECTED VIRTUAL PC IN R1
045174  020167  172154                    CMP     R1,WASSR2       ;DID SR2 CONTAIN VIRTUAL PC AT 15$
045200  001401                            BEQ     16$
045202  104000                            EMT                     ;SR2 NOT TRACKING CORRECTLY
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 6$" = 000663
045204  012767  077406  125076    16$:    MOV     #77406,KIPDR4   ;RESET PDR4 TO 128 BLKS, R/W
045212  012767  077406  125072            MOV     #77406,KIPDR5   ;RESET PDR5 TO 128 BLKS, R/W
045220  012767  021356  133022            MOV     #T0250,MMVEC    ;RESTORE ADDRESS OF NORMAL MEMORY
                                                                  ;MANAGEMENT TRAP ROUTINE TO M.M. VECTOR



                                  ;*******************************************************************************
                                  ;TEST 407        USER ABORT PICKS UP KERNEL SPACE VECTOR
                                  ;*******************************************************************************
045226                            TS407:
045226  004767  002614            1$:     JSR     PC,TOFF         ;TURN OFF T-BIT TRAPPING FOR THIS TEST
045232  005067  132540            2$:     CLR     PSW             ;GO TO KERNEL MODE
045236  012706  001000                    MOV     #KERSTK,KSP     ;SETUP KERNEL STACK PTR.
045242  012767  001400  132370            MOV     #1400,UIPAR0    ;MAP USER PAGE 0 TO 24K
045250  012737  045320  000004            MOV     #4$,@#4         ;LOAD KERNEL VECTOR 4 (LOC.4) WITH 4$
045256  012737  000340  000006            MOV     #340,@#6        ;LOAD VECTOR+2 WITH NEW PSW
045264  012767  140000  132504            MOV     #140000,PSW     ;GO TO USER MODE
045272  012706  000600                    MOV     #USESTK,USP     ;SETUP USER STACK PTR.
045276  012737  045316  000004            MOV     #3$,@#4         ;LOAD USER VECTOR 4 (LOC. 60004) WITH 3$
045304  012737  000340  000006            MOV     #340,@#6        ;LOAD VECTOR+2 WITH NEW PSW
045312  005767  112462                    TST     160000          ;CAUSE TIMEOUT ERROR TRAP TO "4"
                                                                  ;SHOULD PICK UP NEW PC=4$ FROM KERNEL
                                                                  ;LOC. 4, NOT PC=3$ FROM USER LOC. 4 (=60004)
045316                            3$:
045316  104000                            EMT                     ;DID NOT TRAP THRU KERNEL SPACE
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 2$" = 000740
045320  005067  132452            4$:     CLR     PSW             ;BE SURE BACK IN KERNEL MODE
045324  012706  001000                    MOV     #KERSTK,KSP     ;RESTORE KERNEL S.P. IN CASE IT CHANGED
045330  005067  132304                    CLR     UIPAR0          ;REMAP USER PAGE 0 TO 0-4K
045334  012767  140000  132434            MOV     #140000,PSW     ;GO TO USER MODE
045342  012706  000600                    MOV     #USESTK,USP     ;RESTORE USER STACK POINTER
045346  005067  132424                    CLR     PSW             ;GO BACK TO KERNEL MODE
045352  012737  021336  000004            MOV     #T04,@#4        ;RESTORE ADDR. OF NORMAL CPU TRAP HANDLER TO 4
045360  004767  002516                    JSR     PC,TON          ;TURN T-BIT TRAPPING BACK ON
```

```

                                        ;*****************************************************************************
                                        ;TEST 410         RTI IN USER MODE DOES NOT CHANGE PSW
                                        ;*****************************************************************************
 045364                                 TS410:

 045364  012702  170000                         MOV     #170000,R2          ;LOAD "PRESENT & EXPECTED" PSW VALUE INTO R2
 045370  010267  132402                 2$:     MOV     R2,PSW              ;GO TO USER MODE-PRIORITY 0
 045374  012746  000340                         MOV     #340,-(SP)          ;PUT A NEW PSW (PRIORITY=7) ON STACK
 045400  012746  045406                         MOV     #3$,-(SP)           ;PUT NEW PC ON THE STACK
 045404  000002                                 RTI                         ;DO AN RTI FROM USER MODE
 045406  016701  132364                 3$:     MOV     PSW,R1              ;READ NEW PSW INTO R1
 045412  042701  007437                         BIC     #7437,R1            ;MASK OFF COND. CODE, T-BIT, AND UNUSED BITS
 045416  005067  132354                         CLR     PSW                 ;GO BACK TO KERNEL MODE
 045422  020201                                 CMP     R2,R1               ;DID PSW STAY IN USER, PRIORITY=0?
 045424  001401                                 BEQ     TS411
 045426  104000                                 EMT                         ;PSW CHANGED BY AN RTI FROM USER
                                                                            ;FOR A TIGHTER SCOPE LOOP
                                                                            ;REPLACE ERROR CALL WITH
                                                                            ;"BR=2$" = 000760



                                        ;*****************************************************************************
                                        ;TEST 411         KT ERROR SERVICED BEFORE TIMEOUT ERROR
                                        ;*****************************************************************************
 045430                                 TS411:
 045430  012705  077006                 1$:     MOV     #77006,R5           ;LOAD PDR7 DATA INTO R5
 045434  010567  124656                         MOV     R5,KIPDR7           ;MAP PAGE 7 R/W  PLF=176
 045440  012737  045460  000004                 MOV     #3$,@#4             ;SET CPU TRAP VECTOR TO ADDRESS OF 3$
 045446  012737  045462  000250                 MOV     #4$,@#250           ;SET M.M. TRAP VECTOR TO ADDRESS OF 4$
 045454  005237  177700                 2$:     INC     @#177700            ;CAUSE PLF ABORT AND POTENTIAL TIMEOUT
 045460                                 3$:
 045460  104000                                 EMT                         ;TRAPPED THRU CPU TRAP VECTOR BUT SHOULDN'T HAVE
                                                                            ;FOR TIGHTER SCOPE LOOP
                                                                            ;REPLACE ERROR CALL WITH
                                                                            ;"BR 2$" = 000776
 045462  012706  001000                 4$:     MOV     #KERSTK,KSP         ;RESTORE STACK POINTER AFTER TRAPPING
 045466  016767  132100  171656                 MOV     SR0,WASSR0          ;READ STATUS REG.0
 045474  016767  132076  171652         5$:     MOV     SR2,WASSR2          ;READ STATUS REG. 2
 045502  012700  040017                         MOV     #40017,R0           ;LOAD EXPECTED SR0 CONTENTS INTO R0
 045506  020067  171640                         CMP     R0,WASSR0           ;SR0 PLF ERROR BIT SET?
 045512  001401                                 BEQ     6$
 045514  104000                                 EMT                         ;SR0 DIDN'T REPORT PLF ERROR
                                                                            ;FOR TIGHTER SCOPE LOOP
                                                                            ;REPLACE ERROR CALL WITH
                                                                            ;"BR 2$" = 000741
 045516  012701  045454                 6$:     MOV     #2$,R1              ;LOAD EXPECTED SR2 CONTENTS INTO R1
 045522  020167  171626                         CMP     R1,WASSR2           ;WAS SR2 LOCKED BY PLF ABORT?
 045526  001401                                 BEQ     7$
 045530  104000                                 EMT                         ;SR2 DIDN'T LOCK UP VIRTUAL ADDRESS
                                                                            ;FOR TIGHTER SCOPE LOOP
                                                                            ;REPLACE ERROR CALL WITH
                                                                            ;"BR 2$" = 000741
 045532  042767  160000  132032         7$:     BIC     #160000,SR0         ;CLEAR ERROR BITS THAT WERE SET IN SR0
 045540  012737  021336  000004                 MOV     #T04,@#4            ;RESTORE ADDRESS OF NORMAL CPU TRAP HANDLER
 045546  012737  021356  000250                 MOV     #T0250,@#250        ;RESTORE ADDRESS OF NORMAL M.M. TRAP HANDLER
```

```
045554  012767  077406  124534           MOV     #77406,KIPDR7      ;REMAP PAGE 7 TO READ/WRITE PLF=177


                                 ;*******************************************************************************
                                 ;TEST 412        PC & PSW SAVED FOR KT ERROR DURING SERVICE OF TIMEOUT ERROR
                                 ;*******************************************************************************
045562                           TS412:
045562  004767  002260           1$:     JSR     PC,TOFF            ;TURN T-BIT TRAPPING OFF FOR THIS TEST
045566  012767  001400  132052           MOV     #1400,UIPAR3       ;MAP USER PAGE 3 TO 24-28K
045574  012767  001400  132046           MOV     #1400,UIPAR4       ;MAP USER PAGE 4 TO 24-28K
045602  012767  077402  131776           MOV     #77402,UIPDR3      ;MAP USER PAGE 3 READ-ONLY
045610  012767  077406  131772           MOV     #77406,UIPDR4      ;MAP USER PAGE 4 READ/WRITE
045616  012737  045664  000004           MOV     #4$,@#4            ;LOAD ADDRESS OF 4$ IN CPU (TIMEOUT) VECTOR
045624  012737  140017  000006           MOV     #140017,@#6        ;LOAD PSW THAT SHOULD BE PUT ON STACK IN VECTOR+2
045632  012737  045664  000250           MOV     #4$,@#250          ;LOAD ADDRESS OF 4$ IN M.M. TRAP VECTOR
045640  012737  000340  000252           MOV     #340,@#252         ;LOAD A KERNEL PSW IN MMVEC+2
045646  012767  140000  132122   2$:     MOV     #140000,PSW        ;GO TO USER MODE
045654  012706  100002                   MOV     #100002,USP        ;SET USER STACK PTR. SO SECOND PUSH IS IN PG. 3
045660  005737  177700           3$:     TST     @#177700           ;CAUSE TIMEOUT ERROR THAT WILL CAUSE
                                                                    ;R/O ERROR WHEN TRY TO SAVE OLD PC
045664  016601  000002           4$:     MOV     2(KSP),R1          ;PUT PSW SAVED ON KERNEL STACK INTO R1
045670  011603                           MOV     (KSP),R3           ;PUT PC SAVED ON KERNEL STACK INTO R3
045672  016767  131674  171452           MOV     SR0,WASSR0         ;READ THE CONTENTS OF M.M. STATUS REG. 0
045700  016767  131672  171446           MOV     SR2,WASSR2         ;READ THE CONTENTS OF M.M. STATUS REG. 2
045706  042767  160000  131656           BIC     #160000,SR0        ;CLEAR THE ERROR BITS IN SR0
045714  005067  132056                   CLR     PSW                ;BE SURE IN KERNEL MODE
045720  012706  001000                   MOV     #KERSTK,KSP        ;RESTORE KERNEL STACK POINTER
045724  012767  140000  132044           MOV     #140000,PSW        ;GO TO USER MODE
045732  012706  000600                   MOV     #USESTK,USP        ;RESTORE USER STACK POINTER
045736  005067  132034                   CLR     PSW                ;GO BACK TO KERNEL MODE
045742  005067  171412                   CLR     $TMP0              ;CLEAR ERROR INDICATOR
045746  020127  170017                   CMP     R1,#170017         ;WAS THE PSW SAVED THE ONE PICKED UP BY THE
                                                                    ;TIMEOUT TRAP FROM ERRVEC+2?
                                                                    ;VALUE 170017 = PSW FROM LOC. 6 WITH
                                                                    ;PREVIOUS MODE BITS = USER
045752  001402                           BEQ     5$                 ;BRANCH IF YES
045754  005267  171400                   INC     $TMP0              ;WRONG PSW SAVED DURING "DOUBLE ERROR" SEQUENCE
045760  020327  045664           5$:     CMP     R3,#3$+4           ;WAS THE PC AT THE TIME OF THE TIMEOUT ERROR
                                                                    ;SAVED ON THE STACK?
045764  001402                           BEQ     6$                 ;BRANCH IF YES
045766  005267  171366                   INC     $TMP0              ;WRONG PC SAVED DURING TRAP SEQUENCE
045772  026727  171354  020147   6$:     CMP     WASSR0,#20147      ;DID SR0 REPORT - USER, PAGE 3, R/O ABORT?
046000  001402                           BEQ     7$                 ;BRANCH IF YES
046002  005267  171352                   INC     $TMP0              ;SR0 DID NOT REPORT R/O ABORT
046006  026727  171342  045660   7$:     CMP     WASSR2,#3$         ;DID SR2 LOCK UP VIRTUAL ADDR. OF LAST
                                                                    ;INSTRUCTION SUCCESSFULLY FETCHED?
046014  001402                           BEQ     8$                 ;BRANCH IF YES
046016  005267  171336                   INC     $TMP0              ;SR2 DID NOT LOCK UP ADDR. OF TIMEOUT INST.
046022  005767  171332           8$:     TST     $TMP0              ;ANY "ERRORS" DURING TRAP SEQUENCE?
046026  001401                           BEQ     9$
046030  104000                           EMT                        ;THE WRONG PC OR PSW WERE SAVED
                                                                    ;OR SR0 OR SR2 DID NOT REPORT R/O
                                                                    ;ERROR DURING TIMEOUT - KT TRAP
                                                                    ;SEQUENCE
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
```

```
                                                                ;'BR 2$'' = 000710
046032  012737  021336  000004  9$:     MOV     #T04,@#4        ;RESTORE ADDRESS OF NORMAL CPU TRAP HANDLER
046040  012737  000340  000006          MOV     #340,@#6        ;RELOAD ERRVEC+2 WITH KERNEL PSW
046046  012737  021356  000250          MOV     #T0250,@#250    ;RESTORE ADDRESS OF NORMAL M.M. TRAP HANDLER
046054  012767  077406  131524          MOV     #77406,UIPDR3   ;REMAP USER PAGE 3 READ/WRITE
046062  004767  002014                  JSR     PC,TON          ;TURN T-BIT TRAPPING BACK ON
                                ;*************************************************************************
                                ;*
                                ;*      THIS GROUP OF TESTS WILL TEST ALL THE LOGIC ASSOCIATED WITH
                                ;*      THE 'MOVE FROM PREVIOUS'' AND MOVE TO PREVIOUS'' INSTRUCTIONS.
                                ;*
                                ;*************************************************************************


                                ;*************************************************************************
                                ;TEST 413       MOVE FROM PREVIOUS (USER) I-SPACE
                                ;*************************************************************************
046066                          TS413:
046066  005067  124246          1$:     CLR     KIPAR0          ;MAP KERNEL PAGE 0 TO 0-4K
046072  012767  000200  124242          MOV     #200,KIPAR1     ;MAP KERNEL PAGE 1 TO 4-8K
046100  012767  000400  124236          MOV     #400,KIPAR2     ;MAP KERNEL PAGE 2 TO 8-12K
046106  012767  000600  124232          MOV     #600,KIPAR3     ;MAP KERNEL PAGE 3 TO 12-16K
046114  012767  001400  124226          MOV     #1400,KIPAR4    ;MAP KERNEL PAGE 4 TO 24-28K
046122  012767  007600  124226          MOV     #7600,KIPAR7    ;MAP KERNEL PAGE 7 TO THE I/O PAGE
046130  012700  077406                  MOV     #77406,R0       ;MAKE ALL KERNEL I-SPACE PAGES RESIDENT
                                                                ;READ/WRITE, LENGTH 200 BLOCKS
046134  012702  000010                  MOV     #10,R2          ;SET LOOP COUNTER TO 8
046140  012701  172300                  MOV     #KIPDR0,R1      ;PUT ADDRESS OF FIRST PDR IN R1
046144  010021                  2$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
046146  077202                          SOB     R2,2$           ;LOOP TO 2$ UNTIL ALL PDRS LOADED
046150  012702  000010                  MOV     #10,R2          ;SET LOOP COUNTER TO 8
046154  012701  177600                  MOV     #UIPDR0,R1      ;PUT ADDRESS OF FIRST PDR IN R1
046160  010021                  3$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
046162  077202                          SOB     R2,3$           ;LOOP TO 3$ UNTIL ALL PDRS LOADED
046164  012767  000000  131446          MOV     #000,UIPAR0     ;MAP USER I PAGE 0 TO 0-4K
046172  012767  000200  131442          MOV     #200,UIPAR1     ;MAP USER I PAGE 1 TO 4-8K
046200  012767  000400  131436          MOV     #400,UIPAR2     ;MAP USER I PAGE 2 TO 8-12K
046206  012767  000600  131432          MOV     #600,UIPAR3     ;MAP USER I PAGE 3 TO 12-16K
046214  012767  007600  131434          MOV     #7600,UIPAR7    ;MAP USER I PAGE 7 TO THE I/O PAGE
046222                          4$:
046222  012767  077406  124060          MOV     #77406,KIPDR4   ;KERNEL I-SPACE PAGE 4 READ/WRITE
046230  012767  001400  124112          MOV     #1400,KIPAR4    ;MAP KERNEL I PAGE 4 TO 24K
046236  012767  001400  131404          MOV     #1400,UIPAR4    ;MAP USER I PAGE 4 TO 24K
046244  012700  036514                  MOV     #36514,R0       ;LOAD DATA PATTERN INTO R0
046250  010037  100000                  MOV     R0,@#100000     ;LOAD DATA PATTERN INTO PHY 140000
046254  012767  046552  131766          MOV     #23$,MMVEC      ;SET M.M. VECTOR TO 23$
046262  105067  124022                  CLRB    KIPDR4          ;MAKE KERNEL I-SPACE PAGE 4 NON-RESIDENT
                                        ;THE FOLLOWING WILL TEST DSTM=0 MFPI
                                        ;
046266  012767  030340  131502  5$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
046274  006506                  6$:     MFPI    USP             ;PUT USER STACK POINTER ON KERNEL
                                                                ;STACK
046276  022706  001000                  CMP     #KERSTK,KSP     ;WAS SOMETHING PUSHED ON STACK AT 6$
046302  001405                          BEQ     7$              ;BRANCH IF NOTHING WAS PUSHED
046304  012600                          MOV     (KSP)+,R0       ;POP KERNEL STACK INTO R0
046306  012701  000600                  MOV     #USESTK,R1      ;EXPECTING TO GET 700 AS USP
```

```
046312  020001                          CMP     R0,R1             ;DID YOU GET THE RIGHT POINTER?
046314  001401                          BEQ     8$
046316                          7$:
046316  104000                          EMT                       ;WRONG THING WAS PUSHED ON STACK
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 5$" = 000763
046320                          8$:     ;THE FOLLOWING WILL TEST DSTM=1 MFPI.
046320  012700  036514                  MOV     #36514,R0         ;RELOAD DATA PATTERN IN R0
046324  012767  030340  131444  9$:     MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046332  012702  100000                  MOV     #100000,R2        ;LOAD VIRTUAL ADDRESS INTO R2
046336  006512                          MFPI    (R2)              ;READ FROM PHYSICAL 140000
046340  012601                          MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046342  020001                          CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046344  001401                          BEQ     10$
046346  104000                          EMT                       ;WRONG DATA WAS FETCHED
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 9$" = 000766
046350                          10$:    ;THE FOLLOWING WILL TEST DSTM=2 MFPI.
046350  012767  030340  131420  11$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046356  012702  100000                  MOV     #100000,R2        ;LOAD VIRTUAL ADDRESS INTO R2
046362  006522                          MFPI    (R2)+             ;READ FROM PHYSICAL 140000
046364  012601                          MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046366  020001                          CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046370  001401                          BEQ     12$
046372  104000                          EMT                       ;WRONG DATA WAS FETCHED
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 11$" = 000766
046374                          12$:    ;THE FOLLOWING WILL TEST DSTM=3 MFPI.
046374  012767  030340  131374  13$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046402  006537  100000                  MFPI    @#100000          ;READ FROM PHYSICAL 140000
046406  012601                          MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046410  020001                          CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046412  001401                          BEQ     14$
046414  104000                          EMT                       ;WRONG DATA WAS FETCHED
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 13$" = 000767
046416                          14$:    ;THE FOLLOWING WILL TEST DSTM=4 MFPI.
046416  012767  030340  131352  15$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046424  012702  100002                  MOV     #100002,R2        ;LOAD VIRTUAL ADDRESS INTO R2
046430  006542                          MFPI    -(R2)             ;READ FROM PHYSICAL 140000
046432  012601                          MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046434  020001                          CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046436  001401                          BEQ     16$
046440  104000                          EMT                       ;WRONG DATA WAS FETCHED
                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                  ;REPLACE ERROR CALL WITH
                                                                  ;"BR 15$" = 000766
046442                          16$:
                                        ;THE FOLLOWING WILL TEST DSTM=5 MFPI.
                                        ;
046442  012767  030340  131326  17$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046450  012767  100000  170706          MOV     #100000,$TMP2     ;LOAD TEST LOC. VIRT. ADDR INTO LOC. $TMP2
```

```
046456  012702  037366                    MOV     #<$TMP2+2>,R2     ;LOAD ADDR. OF $TMP2+2 INTO R2
046462  006552                            MFPI    @-(R2)            ;READ FROM PHYSICAL 140000
046464  012601                            MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046466  020001                            CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046470  001401                            BEQ     18$
046472  104000                            EMT                       ;WRONG DATA WAS FETCHED
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 17$" = 000763
046474                            18$:    ;THE FOLLOWING WILL TEST DSTM=6 MFPI.
                                          ;
046474  012767  030340  131274    19$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046502  005002                            CLR     R2                ;MAKE REGISTER 2 A ZERO
046504  006562  100000                    MFPI    100000(R2)        ;READ FROM PHYSICAL 140000
046510  012601                            MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046512  020001                            CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046514  001401                            BEQ     20$
046516  104000                            EMT                       ;WRONG DATA WAS FETCHED
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;"BR 19$" = 000766
046520                            20$:    ;THE FOLLOWING WILL TEST DSTM=7 MFPI.
                                          ;
046520  012767  030340  131250    21$:    MOV     #030340,PSW       ;MAKE PREVIOUS MODE USER
046526  012767  100000  170630            MOV     #100000,$TMP2     ;LOAD TEST LOC. V.A. INTO $TMP2
046534  012702  037364                    MOV     #$TMP2,R2         ;LOAD ADDRESS OF $TMP2 INTO R2
046540  006572  000000                    MFPI    @0(R2)            ;USE $TMP2 TO FETCH VIRTUAL
                                                                    ;ADDRESS OF 140000
046544  012601                            MOV     (KSP)+,R1         ;POP KERNEL STACK INTO R1
046546  020001                            CMP     R0,R1             ;WAS DATA FETCHED SAME AS STORED
046550  001401                            BEQ     22$
046552                            23$:
046552  104000                            EMT                       ;WRONG DATA WAS FETCHED
                                                                    ;FOR TIGHTER SCOPE LOOP
```

```
                                                        ;REPLACE ERROR CALL WITH
                                                        ;'BR 21$' = 000762
046554  012767  021356  131466  22$:    MOV     #T0250,MMVEC    ;SET M.M. VECTOR TO NORMAL ROUTINE


```

```
                                        ;*******************************************************************************
                                        ;TEST 414        MOVE TO PREVIOUS (USER) I-SPACE
                                        ;*******************************************************************************
046562                                  TS414:
046562  012767  077406  123520          1$:     MOV     #77406,KIPDR4      ;KERNEL I-SPACE PAGE 4 READ/WRITE
046570  012767  077406  131012                  MOV     #77406,UIPDR4      ;USER I-SPACE PAGE 4 READ/WRITE
046576  012767  001400  123544                  MOV     #1400,KIPAR4       ;MAP KERNEL I PAGE 4 TO 24K
046604  012767  001400  131036                  MOV     #1400,UIPAR4       ;MAP USER I PAGE 4 TO 24K
046612  012767  047304  131430                  MOV     #20$,MMVEC         ;SET M.M. VECTOR TO 20$
                                                ;THE FOLLOWING WILL TEST DSTM=0 MTPI
                                                ;
046620  012767  030340  131150          2$:     MOV     #030340,PSW        ;MAKE PREVIOUS MODE USER
046626  012746  007777                          MOV     #7777,-(KSP)       ;PUSH DATA ON KERNEL STACK
046632  006606                                  MTPI    USP                ;LOAD USER STACK POINTER
046634  006506                                  MFPI    USP                ;READ USER STACK POINTER
046636  012601                                  MOV     (KSP)+,R1          ;POP KERNEL STACK INTO R1
046640  022701  007777                          CMP     #7777,R1           ;WAS USER STACK POINTER CHANGED
046644  001401                                  BEQ     3$
046646  104000                                  EMT                        ;USER STACK POINTER NOT CHANGED
                                                                           ;FOR TIGHTER SCOPE LOOP
                                                                           ;REPLACE ERROR CALL WITH
                                                                           ;"BR 2$" = 000764
046650  012767  030340  131120          3$:     MOV     #030340,PSW        ;MAKE PREVIOUS MODE USER
046656  012746  000600                          MOV     #USESTK,-(KSP)     ;GET READY TO RESTORE USER S. POINT
046662  006606                                  MTPI    USP                ;RESTORE USER STACK POINTER
046664                                  4$:     ;THIS WILL TEST DSTM = 1 MTPI.
046664  012702  100000                          MOV     #100000,R2         ;LOAD VIRTUAL ADDRESS INTO R2
046670  012700  125252                          MOV     #125252,R0         ;LOAD TEST DATA INTO R0
046674  010046                          5$:     MOV     R0,-(KSP)          ;PUSH TEST DATA ON KERNEL STACK
046676  105067  123406                          CLRB    KIPDR4             ;MAKE KERNEL I PAGE 4 NON-RESIDENT
046702  006612                                  MTPI    (R2)               ;LOAD TEST DATA INTO PHYSICAL 140000
046704  112767  000006  123376                  MOVB    #006,KIPDR4        ;MAKE KERNEL PAGE 4 RESIDENT
046712  011201                                  MOV     (R2),R1            ;READ FROM ADDRESS 140000
046714  020001                                  CMP     R0,R1              ;SEE IF DATA WAS STORED AT CORRECT PLACE
046716  001401                                  BEQ     6$
046720  104000                                  EMT                        ;INCORRECT STORE
                                                                           ;FOR TIGHTER SCOPE LOOP
                                                                           ;REPLACE ERROR CALL WITH
                                                                           ;"BR 5$" = 000765
046722                                  6$:     ;THE FOLLOWING WILL TEST DSTM=2 MTPI.
                                                ;
046722  012767  030340  131046                  MOV     #030340,PSW        ;MAKE PREVIOUS MODE USER
046730  012700  125252                          MOV     #125252,R0         ;LOAD TEST DATA INTO R0
046734  012702  100000                          MOV     #100000,R2         ;LOAD VIRTUAL ADDRESS INTO R2
046740  010046                          8$:     MOV     R0,-(KSP)          ;PUSH TEST DATA ON KERNEL STACK
046742  105067  123342                          CLRB    KIPDR4             ;MAKE KERNEL PAGE 4 NON-RESIDENT
046746  006612                                  MTPI    (R2)               ;LOAD TEST DATA INTO PHYSICAL 140000
046750  112767  000006  123332                  MOVB    #006,KIPDR4        ;MAKE KERNEL PAGE 4 RESISENT
046756  013701  100000                          MOV     @#100000,R1        ;READ FROM ADDRESS 140000
046762  020001                                  CMP     R0,R1              ;SEE IF DATA WAS STORED CORRECTLY
046764  001401                                  BEQ     9$
046766  104000                                  EMT                        ;INCORRECT STORE
                                                                           ;FOR TIGHTER SCOPE LOOP
                                                                           ;REPLACE ERROR CALL WITH
                                                                           ;"BR 8$" = 000764
046770                                  9$:     ;THIS WILL TEST DSTM = 3 MTPI.
```

```
046770  012767  030340  131000          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
046776  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
047002  010046                  10$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
047004  105067  123300                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
047010  006637  100000                  MTPI    @#100000        ;LOAD TEST DATA INTO PHYSICAL 140000
047014  112767  000006  123266          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
047022  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
047026  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
047030  001531                          BEQ     TS415
047032  104000                          EMT                     ;
047034  001401                          BEQ     11$
047036  104000                          EMT                     ;INCORRECT STORE
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 10$" = 000763
047040                          11$:    ;THIS WILL TEST DSTM = 4 MTPI.
047040  012767  030340  130730          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
047046  012700  125252                  MOV     #125252,R0      ;LOAD TEST DATA INTO R0
047052  010046                  12$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
047054  012702  100002                  MOV     #100002,R2      ;LOAD VIRTUAL ADDRESS INTO R2
047060  105067  123224                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
047064  006642                          MTPI    -(R2)           ;LOAD TEST DATA INTO PHYSICAL 140000
047066  112767  000006  123214          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
047074  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
047100  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
047102  001401                          BEQ     13$
047104  104000                          EMT                     ;INCORRECT STORE
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 12$" = 000762
047106                          13$:    ;THE FOLLOWING WILL TEST DSTM=5 MTPI.
                                        ;
047106  012767  030340  130662          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
047114  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
047120  012702  037366                  MOV     #<$TMP2+2>,R2   ;LOAD ADDR. OF LOC. $TMP2+2 INTO R2
047124  012767  100000  170232          MOV     #100000,$TMP2   ;LOAD VIRT. ADDR. OF TEST LOC. INTO $TMP2
047132  010046                  14$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
047134  105067  123150                  CLRB    KIPDR4          ;MAKE KERNEL PAGE 4 NON-RESIDENT
047140  006652                          MTPI    @-(R2)          ;LOAD TEST DATA INTO PHYSICAL 140000
047142  112767  000006  123140          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
047150  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
047154  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
047156  001401                          BEQ     15$
047160  104000                          EMT                     ;INCORRECT STORE
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 14$" = 000764
047162                          15$:    ;THIS WILL TEST DSTM = 6 MTPI.
                                        ;
047162  012767  030340  130606          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
047170  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
047174  005002                          CLR     R2              ;MAKE REGISTER 2 ZERO
047176  010046                  16$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
047200  105067  123104                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
047204  006662  100000                  MTPI    100000(R2)      ;LOAD TEST DATA INTO PHYSICAL 140000
047210  112767  000006  123072          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
```

```
047216 013701 100000                    MOV     @#100000,R1     ;READ FROM ADDRESS 140000
047222 020001                           CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
047224 001401                           BEQ     17$
047226 104000                           EMT                     ;INCORRECT STORE
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 16$" = 000763
047230                          17$:    ;THE FOLLOWING WILL TEST DSTM=7 MTPI.
                                        ;
047230 012767 030340 130540             MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
047236 012700 125252                    MOV     #125252,R0      ;LOAD TEST DATA INTO R0
047242 012767 100000 170114             MOV     #100000,$TMP2   ;LOAD VIRT. ADDR. OF TEST LOCATION
                                                                ;INTO LOCATION $TMP2
047250 012702 037364                    MOV     #$TMP2,R2       ;LOAD ADDRESS OF $TMP2 INTO R2
047254 010046                   18$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
047256 105067 123026                    CLRB    KIPDR4          ;MAKE KERNEL PAGE 4 NON-RESIDENT
047262 006672 000000                    MTPI    @0(R2)          ;LOAD TEST DATA INTO PHYSICAL 140000
047266 112767 000006 123014             MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
047274 013701 100000                    MOV     @#100000,R1     ;READ FROM ADDRESS 140000
047300 020001                           CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
047302 001401                           BEQ     19$
047304                          20$:
047304 104000                           EMT                     ;INCORRECT STORE
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;"BR 18$" = 000763
047306 012767 021356 130734     19$:    MOV     #T0250,MMVEC    ;RESTORE M.M. VECTOR TO NORMAL ROUTINE

                                ;**************************************************************************
                                ;TEST 415       MOVE FROM PREVIOUS (KERNEL) I-SPACE TO USER MODE
                                ;**************************************************************************
047314                          TS415:
047314 012700 077406            1$:     MOV     #77406,R0       ;MAKE ALL USER I-SPACE PAGES RESIDENT
                                                                ;READ/WRITE, LENGTH 200 BLOCKS
047320 012702 000010                    MOV     #10,R2          ;SET LOOP COUNTER TO 8
047324 012701 177600                    MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST PDR IN R1
047330 010021                   2$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
047332 077202                           SOB     R2,2$           ;LOOP UNTIL 8 USER PDRS LOADED
047334 012767 140340 130434     3$:     MOV     #140340,PSW     ;GO TO USER MODE FOR THIS TEST
047342 012767 077406 122740             MOV     #77406,KIPDR4   ;KERNEL I-SPACE PAGE 4 READ/WRITE
047350 012767 001400 122772             MOV     #1400,KIPAR4    ;MAP KERNEL I PAGE 4 TO 24K
047356 012767 001400 130264             MOV     #1400,UIPAR4    ;MAP USER I PAGE 4 TO 24K
047364 012700 036514                    MOV     #36514,R0       ;LOAD DATA PATTERN INTO R0
047370 010037 100000                    MOV     R0,@#100000     ;LOAD DATA PATTERN INTO PHY 140000
047374 012702 100000                    MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
                                        ;THE FOLLOWING WILL TEST DSTM=0 MFPI
                                        ;
047400 012767 047676 130642             MOV     #21$,MMVEC      ;SET M.M. VECTOR TO 21$
047406 105067 130176                    CLRB    UIPDR4          ;MAKE USER I-SPACE PAGE 4 NON-RESIDENT
047412 012767 140340 130356             MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047420 006506                   4$:     MFPI    KSP             ;PUT KERNEL STACK POINTER ON USER STACK
047422 022706 000600                    CMP     #USESTK,USP     ;WAS SOMETHING PUSHED ON STACK AT 1$
047426 001405                           BEQ     5$              ;BRANCH IF NOTHING WAS PUSHED
047430 012600                           MOV     (USP)+,R0       ;POP USER STACK INTO R0
```

```
047432  012701  001000                    MOV     #KERSTK,R1       ;EXPECTING 1100 AS KSP
047436  020001                            CMP     R0,R1            ;DID YOU GET THE RIGHT POINTER?
047440  001401                            BEQ     6$
047442                            5$:
047442  104000                            EMT                      ;WRONG THING WAS PUSHED ON STACK
                                                                   ;FOR TIGHTER SCOPE LOOP
                                                                   ;REPLACE ERROR CALL WITH
                                                                   ;"BR 4$" = 000766
047444                            6$:     ;THE FOLLOWING WILL TEST DSTM=1 MFPI.
047444  012767  140340  130324    7$:     MOV     #140340,PSW      ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047452  012700  036514                    MOV     #36514,R0        ;LOAD DATA EXPECTED INTO R0
047456  012702  100000                    MOV     #100000,R2       ;LOAD VIRTUAL ADDRESS INTO R2
047462  006512                            MFPI    (R2)             ;READ FROM PHYSICAL 140000
047464  012601                            MOV     (USP)+,R1        ;POP USER STACK INTO R1
047466  020001                            CMP     R0,R1            ;WAS DATA FETCHED SAME AS STORED
047470  001401                            BEQ     9$
047472  104000                            EMT                      ;WRONG DATA WAS FETCHED
                                                                   ;FOR TIGHTER SCOPE LOOP
                                                                   ;REPLACE ERROR CALL WITH
                                                                   ;"BR 7$" = 000764
                                          ;THE FOLLOWING WILL TEST DSM=2 MFPI.
047474  012767  140340  130274    9$:     MOV     #140340,PSW      ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047502  012702  100000                    MOV     #100000,R2       ;LOAD VIRTUAL ADDRESS INTO R2
047506  006522                            MFPI    (R2)+            ;READ FROM PHYSICAL 140000
047510  012601                            MOV     (USP)+,R1        ;POP USER STACK INTO R1
047512  020001                            CMP     R0,R1            ;WAS DATA FETCHED SAME AS STORED
047514  001401                            BEQ     11$
047516  104000                            EMT                      ;WRONG DATA WAS FETCHED
                                                                   ;FOR TIGHTER SCOPE LOOP
                                                                   ;REPLACE ERROR CALL WITH
                                                                   ;"BR 9$" = 000766
                                          ;THE FOLLOWING WILL TEST DSTM=3 MFPI.
047520  012767  140340  130250    11$:    MOV     #140340,PSW      ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047526  006537  100000                    MFPI    @#100000         ;READ FROM PHYSICAL 140000
047532  012601                            MOV     (USP)+,R1        ;POP USER STACK INTO R1
047534  020001                            CMP     R0,R1            ;WAS DATA FETCHED SAME AS STORED
047536  001401                            BEQ     13$
047540  104000                            EMT                      ;WRONG DATA WAS FETCHED
                                                                   ;FOR TIGHTER SCOPE LOOP
                                                                   ;REPLACE ERROR CALL WITH
                                                                   ;"BR 11$" = 000767
                                          ;THE FOLLOWING WILL TEST DSTM=4 MFPI.
047542  012767  140340  130226    13$:    MOV     #140340,PSW      ;MAKE PREVIOUS MODE DERNEL PRESENT USER
047550  012702  100002                    MOV     #100002,R2       ;LOAD VIRTUAL ADDRESS INTO R2
047554  006542                            MFPI    -(R2)            ;READ FROM PHYSICAL 140000
047556  012601                            MOV     (USP)+,R1        ;POP USER STACK INTO R1
047560  020001                            CMP     R0,R1            ;WAS DATA FETCHED SAME AS STORED
047562  001401                            BEQ     15$
047564  104000                            EMT                      ;WRONG DATA WAS FETCHED
                                                                   ;FOR TIGHTER SCOPE LOOP
                                                                   ;REPLACE ERROR CALL WITH
                                                                   ;"BR 13$" = 000766
                                          ;THE FOLLOWING WILL TEST DSTM=5 MFPI.
                                          ;
047566  012767  140340  130202    15$:    MOV     #140340,PSW      ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047574  012767  100000  167562            MOV     #100000,$TMP2    ;LOAD TEST LOC. VIRT. ADDR INTO LOC. $TMP2
```

```
047602  012702  037366                  MOV     #<$TMP2+2>,R2   ;LOAD ADDRESS OF $TMP2+2 INTO R2
047606  006552                          MFPI    @-(R2)          ;READ FROM PHYSICAL 140000
047610  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
047612  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
047614  001401                          BEQ     17$
047616  104000                          EMT                     ;WRONG DATA WAS FETCHED
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 15$'' = 000763
                                        ;THE FOLLOWING WILL TEST DSTM=6 MFPI.
                                        ;
047620  012767  140340  130150  17$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047626  005002                          CLR     R2              ;MAKE REGISTER 2 A ZERO
047630  006562  100000                  MFPI    100000(R2)      ;READ FROM PHYSICAL 140000
047634  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
047636  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
047640  001401                          BEQ     19$
047642  104000                          EMT                     ;WRONG DATA WAS FETCHED
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 17$'' = 000766
                                        ;THE FOLLOWING WILL TEST DSTM=7 MFPI.
                                        ;
047644  012767  140340  130124  19$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
047652  012767  100000  167504          MOV     #100000,$TMP2   ;LOAD TEST LOC. VIRT. ADDR. INTO $TMP2
047660  012702  037364                  MOV     #$TMP2,R2       ;LOAD ADDRESS OF $TMP2 INTO R2
047664  006572  000000                  MFPI    @0(R2)          ;READ FROM PHYSICAL 140000
047670  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
047672  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
047674  001401                          BEQ     20$
047676                          21$:
047676  104000                          EMT                     ;WRONG DATA WAS FETCHED
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 19$'' = 000762
047700  012767  021356  130342  20$:    MOV     #T0250,MMVEC    ;SET M.M. VECTOR TO NORMAL ROUTINE
047706  012767  000340  130062          MOV     #00340,PSW      ;GO BACK TO KERNEL MODE, PREVIOUS KERNEL

                                ;*****************************************************************************
                                ;TEST 416        MOVE FROM/TO D-SPACE = MOVE FROM/TO I-SPACE
                                ;*****************************************************************************
047714                          TS416:
047714  012767  030340  130054  1$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE=USER,CURRENT=KERNEL
047722  106506                          MFPD    USP             ;MFPD SHOULD ACT LIKE MFPI PUTTING
                                                                ;USER STACK POINTER ON THE KERNEL STACK
047724  022706  001000                  CMP     #KERSTK,KSP     ;WAS SOMETHING PUSHED ON KERNEL STACK?
047730  001405                          BEQ     2$              ;BRANCH IF NO
047732  012600                          MOV     (KSP)+,R0       ;POP KERNEL STACK INTO R0
047734  012701  000600                  MOV     #USESTK,R1      ;EXPECTING TO GET 700 AS USP
047740  020001                          CMP     R0,R1           ;DID GET RIGHT POINTER VALUE?
047742  001401                          BEQ     4$
047744                          2$:
047744  104000                          EMT                     ;WRONG THING WAS PUSHED ON STACK
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 1$'' = 000763
```

```
047746  012746  007777   4$:     MOV     #7777,-(KSP)     ;PUSH DATA ON KERNEL STACK
047752  106606                   MTPD    USP              ;LOAD THE USER STACK POINTER
047754  106506                   MFPD    USP              ;READ USER STACK POINTER
047756  012601                   MOV     (KSP)+,R1        ;POP KERNEL STACK INTO R1
047760  022701  007777           CMP     #7777,R1         ;WAS USER STACK POINTER CHANGED?
047764  001401                   BEQ     5$
047766  104000                   EMT                      ;USER STACK POINTER NOT CHANGED
                                                          ;FOR TIGHTER SCOPE LOOP
                                                          ;REPLACE ERROR CALL WITH
                                                          ;'BR 4$'' = 000767
047770  012746  000600   5$:     MOV     #USESTK,-(KSP)   ;GET READY TO RESTORE USER STK. PTR.
047774  106606                   MTPD    USP              ;RESTORE USER STACK POINTER

                         ;*************************************************************************
                         ;TEST 417        MOVE FROM PREVIOUS I=SPACE (PREVIOUS=CURRENT=KERNEL)
                         ;*************************************************************************
047776                   TS417:
047776  005037  177776   1$:     CLR     @#PSW            ;SET PREVIOUS = CURRENT = KERNEL
050002  012700  001000           MOV     #KERSTK,R0       ;SETUP VALUE FOR STACK POINTER
050006  010006                   MOV     R0,KSP           ;LOAD STACK POINTER
050010  006506                   MFPI    KSP              ;THE VALUE "STACK" SHOULD BE PUSHED
                                                          ;BEFORE BEING DECREMENTED
050012  011601                   MOV     (KSP),R1         ;READ DATA WHICH WAS PUSHED
050014  020001                   CMP     R0,R1            ;WAS THE ORIGINAL VALUE OF THE
                                                          ;STACK POINTER PUSHED?
050016  001401                   BEQ     2$
050020  104000                   EMT                      ;MFPI FETCHED WRONG DATA
                                                          ;FOR TIGHTER SCOPE LOOP
                                                          ;REPLACE ERROR CALL WITH
                                                          ;'BR 1$'' = 000766
050022  005740           2$:     TST     -(R0)            ;SETUP EXPECTED STACK POINTER VALUE
050024  020600                   CMP     KSP,R0           ;WAS THE STACK POINTER DECREMENTED?
050026  001401                   BEQ     3$
050030  104000                   EMT                      ;STACK NOT PUSHED BY THE MFPI
                                                          ;FOR TIGHTER SCOPE LOOP
                                                          ;REPLACE ERROR CALL WITH
                                                          ;'BR 1$'' = 000762
050032  012706  001000   3$:     MOV     #KERSTK,KSP      ;RESTORE STACK POINTER
050036  005067  127530           CLR     SR0              ;TURN OFF MEMORY MANAGEMENT UNIT
050042  000167  000472           JMP     EXATST           ;GET OVER SUBROUTINES TO EXTENDED ADRS TESTS
```

```
                                          .SBTTL  *****   SUBROUTINES USED BY THIS PROGRAM   *****

                                          .SBTTL  TURN OFF T-BIT AND SAVE CURRENT PSW
                                          ;****************************************************************************
                                          ;*
                                          ;*      THIS SUBROUTINE IS USED TO TURN OFF THE TRACE TRAP BIT IN THE PSW
                                          ;*      IF IT IS ON.  THE PROCESSOR STATUS IS SAVED IN ''TBITPS'' SO THAT
                                          ;*      THE PSW CAN BE RESTORED TO ITS PREVIOUS CONDITION WHEN CONDITIONS
                                          ;*      WARRANT T-BIT TRAPPING.
                                          ;*
                                          ;****************************************************************************
050046  036727  127724  000020    TOFF:   BIT     PSW,#TBIT         ;IS THE T-BIT SET IN THE PSW?
050054  001411                            BEQ     1$                ;EXIT IF NO
050056  016746  127714                    MOV     PSW,-(SP)         ;PUSH PRESENT PSW ON THE STACK
050062  011667  167270                    MOV     (SP),TBITPS       ;ALSO SAVE IT IN ''TBITPS'' FOR
                                                                    ;RESTORING LATER
050066  042716  000020                    BIC     #TBIT,(SP)        ;CLEAR THE T-BIT (BIT 4) IN THE PSW
050072  012746  050100                    MOV     #1$,-(SP)         ;PUSH PC OF ''RTS'' ON STACK
050076  000006                            RTT                       ;''RETURN'' TO 1$ WITH T-BIT OFF
050100  000207                    1$:     RTS     PC                ;RETURN TO PROGRAM

                                          .SBTTL  TURN ON T-BIT AND RESTORE PREVIOUS PSW
                                          ;****************************************************************************
                                          ;*
                                          ;*      THIS SUBROUTINE IS USED TO RESTORE THE PROCESSOR STATUS TO ITS
                                          ;*      PREVIOUS CONDITION BY RESTORING THE ''T-BIT PSW'' SAVED BY THE
                                          ;*      ''TOFF'' SUBROUTINE IN THE ''TBITPS'' LOCATION.
                                          ;*
                                          ;****************************************************************************
050102  036727  167250  000020    TON:    BIT     TBITPS,#TBIT      ;WAS T-BIT ON IN THE PREVIOUS PSW?
050110  001410                            BEQ     1$                ;EXIT IF NO
050112  016746  167240                    MOV     TBITPS,-(SP)      ;PUSH PREVIOUS PSW ON THE STACK
050116  012767  000340  167232            MOV     #340,TBITPS       ;RESET THE ''TBITPS'' LOCATION
050124  012746  050132                    MOV     #1$,-(SP)         ;PUSH PC OF ''RTS'' ON STACK
050130  000006                            RTT                       ;''RETURN'' TO 1$ WITH T-BIT RESTORED
050132  000207                    1$:     RTS     PC                ;RETURN TO PROGRAM

                                          .SBTTL  SET ALL WRITEABLE BITS IN ALL PAR/PDR'S
                                          ;****************************************************************************
                                          ;*
                                          ;*      THIS SUBROUTINE IS USED BY THE PAR/PDR DUAL ADDRESSING TEST
                                          ;*      TO SET ALL WRITEABLE BITS IN ALL KERNEL AND USE PAR'S AND
                                          ;*      PDR'S TO A 1.  THE ''INITIAL STATE'' OF HAVING ALL BITS=1 IS
                                          ;*      USED TO SEE THAT ONLY ONE REGISTER IS CLEARED IN RESPONSE TO
                                          ;*      A SINGLE PAR OR PDR ADDRESS.
                                          ;*
                                          ;****************************************************************************

050134  012702  000010            SETREG: MOV     #10,R2            ;LOAD LOOP COUNTER WITH AN 8
050140  012701  172300                    MOV     #KIPDR0,R1        ;LOAD ADDRESS OF FIRST PDR INTO R1
050144  012721  177777            1$:     MOV     #-1,(R1)+         ;SET BITS IN KERNEL PDR TO 1
050150  077203                            SOB     R2,1$             ;LOOP TO 1$ UNTIL ALL KERNEL PDR'S LOADED
050152  012702  000010                    MOV     #10,R2            ;LOAD LOOP COUNTER WITH AN 8
050156  012701  172340                    MOV     #KIPAR0,R1        ;LOAD ADDRESS OF FIRST PAR INTO R1
050162  012721  177777            2$:     MOV     #-1,(R1)+         ;SET BITS IN A KERNEL PAR TO 1
050166  077203                            SOB     R2,2$             ;LOOP TO 2$ UNTIL ALL KERNEL PAR'S LOADED
```

```
050170  012702  000010                  MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
050174  012701  177600                  MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST PDR INTO R1
050200  012721  177777          3$:     MOV     #-1,(R1)+       ;SET BITS IN A USER PDR TO 1
050204  077203                          SOB     R2,3$           ;LOOP TO 3$ UNTIL ALL USER PDR'S LOADED
050206  012702  000010                  MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
050212  012701  177640                  MOV     #UIPAR0,R1      ;LOAD ADDRESS OF FIRST PAR INTO R1
050216  012721  177777          4$:     MOV     #-1,(R1)+       ;SET BITS IN A USER PAR TO 1
050222  077203                          SOB     R2,4$           ;LOOP TO 4$ UNTIL ALL USER PAR'S LOADED
050224  000207                          RTS     PC              ;RETURN TO TEST

                                .SBTTL  READ & COMPARE KERNEL & USER PAR/PDR'S
                                ;*******************************************************************************
                                ;*
                                ;*      THIS SUBROUTINE IS USED BY PAR/PDR DUAL ADDRESSING TEST TO
                                ;*      READ ALL THE PAR'S AND PDR'S TO SEE THAT ONLY ONE REGISTER
                                ;*      WAS CLEARED IN RESPONSE TO A SINGLE PAR OR PDR ADDRESS.
                                ;*      ANY FAILURES FOUND BY THE PAR/PDR DUAL ADDRESSING TEST WILL
                                ;*      BE REPORTED BY THIS SUBROUTINE.
                                ;*
                                ;*******************************************************************************
050226                          CMPREG:
050226  012701  172300                  MOV     #KIPDR0,R1      ;LOAD ADDRESS OF FIRST KERNEL PDR IN R1
050232  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
050236  012705  077416                  MOV     #77416,R5       ;PUT EXPECTED PDR CONTENTS IN R5
050242  021105                  1$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
050244  001403                          BEQ     2$              ;BRANCH IF YES
050246  020100                          CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
050250  001401                          BEQ     2$
050252  104000                          EMT                     ;A PDR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PRD
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;AN "RTS PC" = 000207
050254  062701  000002          2$:     ADD     #2,R1           ;FORM NEXT ADDRESS
050260  077410                          SOB     R4,1$           ;LOOP TO 1$ UNTIL ALL KERNEL PDR'S CHECKED
050262  012701  172340                  MOV     #KIPAR0,R1      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R1
050266  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
                                                                ;****F11 CHANGE**** FROM #7777 TO #177777
050272  012705  177777                  MOV     #177777,R5      ;PUT EXPECTED PAR CONTENTS IN R5
050276  021105                  3$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
050300  001403                          BEQ     4$              ;BRANCH IF YES
050302  020100                          CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
050304  001401                          BEQ     4$
050306  104000                          EMT                     ;A PAR WAS EFFECTED BY CLEARING A DIFFENENT PAR/PDR
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;AN "RTS PC" = 000207
050310  062701  000002          4$:     ADD     #2,R1           ;FORM NEXT ADDRESS
050314  077410                          SOB     R4,3$           ;LOOP TO 3$ UNTIL ALL KERNEL PAR'S CHECKED
050316  012701  177600                  MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST USER PDR IN R1
050322  012704  000010                  MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
050326  012705  077416                  MOV     #77416,R5       ;PUT EXPECTED PDR CONTENTS IN R5
050332  021105                  5$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
050334  001403                          BEQ     6$              ;BRANCH IF YES
050336  020100                          CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
050340  001401                          BEQ     6$
050342  104000                          EMT                     ;A PDR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PDR
```

```
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;AN "RTS PC" = 000207
050344  062701  000002          6$:     ADD     #2,R1               ;FORM NEXT ADDRESS
050350  077410                          SOB     R4,5$               ;LOOP TO 5$ UNTIL ALL USER PDR'S CHECKED
050352  012701  177640                  MOV     #UIPAR0,R1          ;LOAD ADDRESS OF FIRST USER PAR IN R1
050356  012704  000010                  MOV     #10,R4              ;LOAD LOOP COUNTER WITH AN 8
                                                                    ;****F11 CHANGE**** FROM #7777 TO #177777
050362  012705  177777                  MOV     #177777,R5          ;PUT EXPECTED PAR CONTENTS IN R5
050366  021105                  7$:     CMP     (R1),R5             ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
050370  001403                          BEQ     8$                  ;BRANCH IF YES
050372  020100                          CMP     R1,R0               ;WAS IT THE REG. THAT WAS CLEARED?
050374  001401                          BEQ     8$
050376  104000                          EMT                         ;A PAR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PDR
                                                                    ;FOR TIGHTER SCOPE LOOP
                                                                    ;REPLACE ERROR CALL WITH
                                                                    ;AN "RTS PC" = 000207
050400  062701  000002          8$:     ADD     #2,R1               ;FORM NEXT ADDRESS
050404  077410                          SOB     R4,7$               ;LOOP TO 7$ UNTIL ALL USER PAR'S CHECKED
050406  000207                          RTS     PC                  ;RETURN TO TEST

                                .SBTTL  INHIBIT 'RESETS' WHILE UNDER APT
                                ;*************************************************************************
                                ;*
                                ;*      THIS SUBROUTINE CONTROLS THE USAGE OF RESET INST'S WHILE
                                ;*      RUNNING UNDER APT. RESETS ARE ALLOWED DURING THE FIRST
                                ;*      PASS OF THE DIAGNOSTIC.
                                ;*
050410  126727  130404  000001  CHKAPT: CMPB    $ENV,#1             ;ARE WE RUNNING UNDER APT?
050416  001003                          BNE     1$                  ;NO BRANCH
050420  005767  130362                  TST     $PASS               ;IS THIS THE FIRST PASS?
050424  001002                          BNE     RETA                ;NO BRANCH
050426  062705  000002          1$:     ADD     #2,R5               ;BUMP RETURN ADDRESS FOR NORMAL TESTING
050432  000205                  RETA:   RTS     R5                  ;RETURN


                                .SBTTL  ERROR ROUTINE FOR MEMORY MANAGEMENT TEST
                                ;*************************************************************************
                                ;*      THIS IS THE ONLY ERROR REPORT FOR ALL THE MMU TESTS
                                ;*************************************************************************

050434  012737  000002  001002  ERROR3: MOV     #2,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
050442  012767  000001  130330          MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
050450  032737  000001  001020          BIT     #1,@#$ENV                       ;UNDER APT ?
050456  001004                          BNE     MMUHLT
050460  012700  050472                  MOV     #MMUMSG,R0
050464  004767  062640                  JSR     PC,TYPE
050470  000777                  MMUHLT: BR      .                   ;STAY IN LOOP

050472  040506  046111  042105  MMUMSG: .ASCIZ  /FAILED DURING MMU TESTING/<12><15>
050500  042040  051125  047111
050506  020107  046515  020125
050514  042524  052123  047111
050522  005107  000015

                                .EVEN
```

```
                                            ;******************* ************************************************
                                            ;
                                            ;               STORAGE AREAS FOR THE FOLLOWING FEW TESTS
                                            ;
                                            ;********************************************************************
050526  000000                              TMP:    .WORD   0
050530  000000                              TMP1:   .WORD   0
050532  000000                              MEM:    .WORD   0
050534  000002                              CNTR:   .BLKW   2
                                            ;*****************************************************************************************
                                            ;TEST 420        TEST ADDRESS BITS 17-21
                                            ;*****************************************************************************************
050540                                      TS420:
                                            ;*****************************************************************
                                            ;
                                            ;               THIS TEST WILL DETERMINE WHETHER THE KDF11-B CAN
                                            ;               DRIVE ADDRESS BITS 17-21.
                                            ;
                                            ;******* *********************************************************
050540  012767  177600  121610  EXATST: MOV     #177600,KIPAR7                  ;I O PAGE
050546  012737  000007  001004          MOV     #7,@#$TESTN
050554  012737  051332  000030          MOV     #ERRORA,@#30
050562  000244                          CLZ                                     ;CLR THE Z BIT
050564  032777  000200  150612          BIT     #200,@SWR                       ;WANT TO TEST THIS ?
050572  001001                          BNE     ESR3
050574  000470                          BR      NEXT
050576  016767  127202  177722  ESR3:   MOV     ERRVEC,TMP
050604  016767  127176  177716          MOV     ERRVEC+2,TMP1
050612  012767  050714  127164          MOV     #ESR0,ERRVEC                    ;PREPARE FOR NEW
050620  012767  000340  127160          MOV     #340,ERRVEC+2                   ;INTERRUPT
050626  012767  004000  121514          MOV     #4000,KIPAR4                    ;SET FOR ADRS BIT 17
050634  012767  000001  126730          MOV     #BIT0,SR0
050642  012767  000020  121646          MOV     #BIT4,SR3                       ;TURN 22 BIT ADRSG AND
                                                                                ;MEMORY MANG ON
050650  012767  000000  177656          MOV     #0,CNTR
050656  026727  177652  000005  ESR1:   CMP     CNTR,#5
050664  001420                          BEQ     ESR2
050666  013737  100000  100000          MOV     @#100000,@#100000               ;TRY ADRSG IT
050674  000240                          NOP
050676  012767  000000  126666          MOV     #0,SR0
050704  012767  000000  121604          MOV     #0,SR3                          ;TURN OFF MM AND 22 BIT
050712  104000                          EMT                                     ;ERROR
050714  006367  121430          ESR0:   ASL     KIPAR4                          ;SHOULD TIME OUT
050720  005267  177610                  INC     CNTR                            ;SHIFT 1 TO THE LEFT
050724  000754                          BR      ESR1
050726  016767  177574  127050  ESR2:   MOV     TMP,ERRVEC
050734  016767  177570  127044          MOV     TMP1,ERRVEC+2                   ;RESTORE
050742  012767  000000  126622          MOV     #0,SR0
050750  012767  000000  121540          MOV     #0,SR3                          ;TURN OFF MM AND 22 BIT
```

```
                                        ;*******************************************************************************
                                        ;TEST 421        TEST ADDRESS BIT 16
                                        ;*******************************************************************************
050756                                  TS421:
                                        ;***********************************************************************
                                        ;
                                        ;       THIS TEST WILL CHECK TO SEE IF THE 17TH BIT (16) OF AN 18 BIT
                                        ;       ADDRESS CAN BE DRIVEN.
                                        ;
                                        ;***************************************************************************

050756  012767  000001  126606  NEXT:   MOV     #BIT0,SR0                       ;TURN ON MEM MANG
050764  016767  127014  177534          MOV     ERRVEC,TMP
050772  016767  127010  177530          MOV     ERRVEC+2,TMP1
051000  012767  051062  126776          MOV     #ESR4,ERRVEC
051006  012767  000340  126772          MOV     #340,ERRVEC+2
051014  012737  123456  000000          MOV     #123456,@#0                     ;MODIFY LOCATION ZERO
051022  012767  002000  121320          MOV     #2000,KIPAR4
051030  013737  100000  100000          MOV     @#100000,@#100000
051036  023727  000000  123456          CMP     @#0,#123456                     ;CHANGED ?
051044  001407                          BEQ     ESR5
051046  012767  000000  126516          MOV     #0,SR0
051054  012767  000000  121434          MOV     #0,SR3                  ;TURN OFF MM AND 22 BIT
051062  104000                  ESR4:   EMT                                     ;LOC 0 WAS ALTERED
051064  005067  126502          ESR5:   CLR     SR0                             ;TURN OFF MM
051070  016767  177432  126706          MOV     TMP,ERRVEC
051076  016767  177426  126702          MOV     TMP1,ERRVEC+2                   ;RESTORE VECTORS



                                        ;*******************************************************************************
                                        ;TEST 422        TEST PARITY ERROR DETECTION LOGIC
                                        ;*******************************************************************************
051104                                  TS422:

                                        ;*******************************************************************
                                        ;
                                        ;       THIS TEST WILL USE THE MEMORY PARITY CSR TO GENERATE
                                        ;       A PARITY ERROR THAT THE CPU SHOULD DETECT.
                                        ;
                                        ;**********************************************************************


051104  000244                          CLZ                             ;CLR THE Z BIT
051106  032777  001000  150270          BIT     #BIT9,@SWR              ;TEST THIS ?
051114  001002                          BNE     ESR57
051116  000167  000430                  JMP     Q22TST                  ;NO

051122  012767  000001  000370  ESR57:  MOV     #1,ERRORE               ;DUMMY BIT
051130  016767  126650  177376          MOV     ERRVEC,CNTR
051136  016767  126644  177372          MOV     ERRVEC+2,CNTR+2         ;SAVE VECTORS
051144  012767  051162  126632          MOV     #2$,ERRVEC              ;NEW TIME OUT VECOTR
051152  012704  172100                  MOV     #172100,R4              ;PLACE TO START
051156  005714                  1$:     TST     (R4)                    ;IS IT THERE
051160  000407                          BR      ESR7
051162  022704  172136          2$:     CMP     #172136,R4              ;TOP YET ?
```

```
051166  100403                                  BMI     ESR6
051170  062704  000002                          ADD     #2,R4
051174  000770                                  BR      1$
051176  104000                          ESR6:   EMT                             ;COULD NOT FIND MEM PAR CSR
051200  016767  126710  177320          ESR7:   MOV     114,TMP                 ;SAVE CONTENTS OF 114
051206  012767  051254  126700                  MOV     #ESR8,114               ;PARITY TRAP VECTOR
051214  012714  000005                          MOV     #5,(R4)                 ;BITS 0,2 IN MEM CSR
051220  010167  177306                          MOV     R1,MEM
051224  042714  000004                          BIC     #4,(R4)                 ;TURN OFF WRT WRG PAR BIT
051230  016701  177276                          MOV     MEM,R1                  ;SOME DATA  TO MEMORY FROM
051234  012714  000000                  ESR9:   MOV     #0,(R4)                 ;TURN OFF P ERR
051240  010167  177266                          MOV     R1,MEM
051244  016767  177256  126642                  MOV     TMP,114                 ;RESTORE VECTOR CONTENTS
                                                                                ;CPU, SHOULD SET A PAR ERR
051252  104000                                  EMT
051254  032714  100000                  ESR8:   BIT     #BIT15,(R4)             ;SEE IF BIT 15 SET IN MEM CSR
051260  001765                                  BEQ     ESR9                    ;Z IS SET IF AN ERROR
051262  005000                                  CLR     R0
051264  005200                          ESR19:  INC     R0                      ;TIME FOR LED TO BE SEEN
051266  005700                                  TST     R0
051270  100375                                  BPL     ESR19
051272  012714  000000                          MOV     #0,(R4)
051276  010167  177230                          MOV     R1,MEM
051302  016767  177220  126604                  MOV     TMP,114
051310  016767  177220  126466                  MOV     CNTR,ERRVEC
051316  016767  177214  126462                  MOV     CNTR+2,ERRVEC+2
051324  005067  000170                          CLR     ERRORE                  ;RESTORE DUMMY BIT
051330  000510                                  BR      Q22TST                  ;GO FIND A Q22BE
                                        ;**************************************************************************

051332  012737  000007  001002          ERRORA: MOV     #7,@#$FATAL
051340  012767  000001  127432                  MOV     #1,$MSGTY
051346  032737  000001  001020                  BIT     #1,@#$ENV                       ;UNDER APT ?
051354  001013                                  BNE     EXADHT
051356  026727  000136  000001                  CMP     ERRORE,#1                       ;DUMMY BIT THERE  ?
051364  001003                                  BNE     ESR34
051366  012700  051451                          MOV     #CSRMSG,R0
051372  000402                                  BR      ESR34+4
051374  012700  051406                  ESR34:  MOV     #EXTMSG,R0
051400  004767  061724                          JSR     PC,TYPE
051404  000777                          EXADHT: BR      .


051406  040506  046111  042105          EXTMSG: .ASCIZ  /FAILED DURING EXTENDED ADRS TEST/<12><15>
051414  042040  051125  047111
051422  020107  054105  042524
051430  042116  042105  040440
051436  051104  020123  042524
051444  052123  006412     000

051451     106  044501  042514          CSRMSG: .ASCIZ  /FAILED MEM PARITY ERROR DETECT TEST/<12><15>
051456  020104  042515  020115
051464  040520  044522  054524
051472  042440  051122  051117
051500  042040  052105  041505
051506  020124  042524  052123
```

```
051514  006412         000
               051520                          .EVEN
051520  000000                                 ERRORE: .WORD   0               ;DUMMY BIT
                                               ;*****************************************************************
051522  000510                                 VECT1:  .WORD   510             ;FIRST DEV VECTOR Q22BE
051524  000000                                 DEVECT: .WORD   0
051526  170000                                 DEV1:   .WORD   170000                  ;FIRST DEV ADRS Q22BE
051530  000000                                 DEVADR: .WORD   0
051532  000000                                 CSR1:   .WORD   0
051534  000000                                 CSR2:   .WORD   0
051536  000000                                 BA:     .WORD   0
051540  000000                                 WC:     .WORD   0
051542  000000                                 DATA:   .WORD   0
051544  000000                                 LATCNT: .WORD   0
051546  000000                                 MVLCNT: .WORD   0
051550  000000                                 SIMGOA: .WORD   0
                                               ;*************************************************************************************
                                               ;TEST 423        SEE IF A Q22BE(QBE) IS THERE
                                               ;*************************************************************************************
051552                                         TS423:
                                               ;**********************************************************
                                               ;
                                               ;        ROUTINE TO SIZE FOR THE Q22BE(QBE) DEVICE ADDRESS
                                               ;        WE WILL LOOK FOR A Q22BE(QBE). IF IT ISN'T THERE
                                               ;        WE WILL CAUSE AN ERROR VIA THE EMT INSTRUCTION.
                                               ;
                                               ;**********************************************************
051552  000244                                 Q22TST: CLZ
051554  032777  000100  147622                         BIT     #BIT6,@SWR
051562  001002                                         BNE     1$
051564  000167  001476                                 JMP     BDVTST
051570  012737  000011  001004                 1$:     MOV     #11,@#$TESTN            ;TEST NUM IN MAILBOX
051576  012737  053160  000030                         MOV     #ERRORB,@#30            ;SET UP FOR CORRECT EMT
051604  005067  125762                                 CLR     SR0
051610  016767  126170  176714                         MOV     ERRVEC,MEM              ;STORE ERRVEC CONTENTS
051616  012767  052006  126160                         MOV     #ESR99,ERRVEC
051624  016767  177676  177676                         MOV     DEV1,DEVADR
051632  016767  177664  177664                         MOV     VECT1,DEVECT
051640  005777  177664                         ESR11:  TST     @DEVADR         ;SEE IF IT RESPONDS
051644  016767  177660  177660                         MOV     DEVADR,CSR1
051652  016767  177654  177654                         MOV     CSR1,CSR2
051660  062767  000002  177646                         ADD     #2,CSR2                 ;YES IT DID
051666  016767  177640  177642                         MOV     CSR1,BA
051674  062767  000004  177634                         ADD     #4,BA
051702  016767  177624  177630                         MOV     CSR1,WC
051710  062767  000006  177622                         ADD     #6,WC
```

```
051716  016767  177610  177616           MOV     CSR1,DATA
051724  062767  000010  177610           ADD     #10,DATA
051732  016767  177574  177604           MOV     CSR1,LATCNT
051740  062767  000012  177576           ADD     #12,LATCNT
051746  016767  177560  177572           MOV     CSR1,MVLCNT
051754  062767  000014  177564           ADD     #14,MVLCNT
051762  016767  177544  177560           MOV     CSR1,SIMGOA
051770  062767  000016  177552           ADD     #16,SIMGOA
051776  016767  176530  126000           MOV     MEM,ERRVEC                    ;RESTORE ERROR VECTOR
052004  000413                           BR      ESR98
052006  062767  000020  177514  ESR99:   ADD     #20,DEVADR
052014  062767  000004  177502           ADD     #4,DEVECT
052022  026727  177476  000550           CMP     DEVECT,#550
052030  001303                           BNE     ESR11                         ;GO TRY ANOTHER ADRS
052032  104000                           EMT
052034  016700  177464          ESR98:   MOV     DEVECT,R0
052040  062700  000002                   ADD     #2,R0
052044  012710  000340                   MOV     #340,(R0)


                                ;*****************************************************************************
                                ;TEST 424        USE Q22BE(QBE) TO ALTER THE INTERRUPT LEVEL BITS
                                ;*****************************************************************************
052050                          TS424:

                                ;*************************************************************
                                ;
                                ;               GENERATE A Q22BE(QBE) SOFTWARE INTR AT LEVEL 4.
                                ;               THE FOLLOWING CODE WILL USE THE Q22(QBE) BUS EXERCISER
                                ;               TO GENERATE INTERRUPTS THAT THE CPU SHOULD EITHER
                                ;               HONOR OR IGNORE DEPENDING ON THE INTR LEVEL BITS.
                                ;
                                ;*******************************************************************


052050  012777  052114  177446           MOV     #ESR21,@DEVECT                ;INTERRUPT TO ESR21
052056  012777  000001  177446           MOV     #1,@CSR1
052064  106427  000140                   MTPS    #140                 ;CPU AT LEVEL THREE
052070  012777  000003  177436           MOV     #3,@CSR2
052076  000240                           NOP
052100  012777  000002  177426           MOV     #2,@CSR2
052106  000240                           NOP
052110  000240                           NOP
052112  104000                           EMT
052114  012777  052150  177402  ESR21:   MOV     #ESR22,@DEVECT                ;CHANGE INTR VECTOR
052122  106427  000200                   MTPS    #200
052126  012777  000003  177400           MOV     #3,@CSR2
052134  000240                           NOP
052136  012777  000000  177370           MOV     #0,@CSR2                      ;INTR LEVEL IS 4
052144  000240                           NOP                                   ;INTR SHOULD NOT HONORED
052146  000401                           BR      ESR23
052150  104000                  ESR22:   EMT                                   ;INTR HONORED A ERROR
                                ;*******************************************************************
                                ;               GENERATE AN INTERRUPT AT LEVEL FIVE
                                ;*********************************************************************
```

```
052152  012777  052214  177344  ESR23:  MOV     #ESR24,@DEVECT
052160  106427  000200                  MTPS    #200                    ;CPU AT LEVEL FOUR
052164  012777  000001  177340          MOV     #1,@CSR1                ;TRY TO CAUSE AN INTERRUPT
052172  012777  000007  177334          MOV     #7,@CSR2                ;AT LEVEL FIVE
052200  000240                          NOP
052202  012777  000006  177324          MOV     #6,@CSR2                ;CLRS GO, SETS DONE
052210  000240                          NOP
052212  104000                          EMT                             ;INTR DID NOT HAPPEN
052214  012777  052250  177302  ESR24:  MOV     #ESR27,@DEVECT          ;ALTER INTR VECTOR
052222  106427  000240                  MTPS    #240                    ;CHANGE LEVEL TO FIVE
052226  012777  000007  177300          MOV     #7,@CSR2
052234  000240                          NOP
052236  012777  000000  177270          MOV     #0,@CSR2
052244  000240                          NOP
052246  000401                          BR      ESR28
052250  104000                  ESR27:  EMT                             ;IF HERE, AN ERROR
                                ;*******************************************************
                                ;       TRY INTERRUPT AT LEVEL SIX
                                ;*******************************************************
052252  012777  052310  177244  ESR28:  MOV     #ESR29,@DEVECT
052260  012777  000001  177244          MOV     #1,@CSR1                ;INTR RQST BITS TO 6
052266  012777  000013  177240          MOV     #13,@CSR2
052274  000240                          NOP
052276  012777  000012  177230          MOV     #12,@CSR2               ;INTR SHOULD BE HONORED
052304  000240                          NOP
052306  104000                          EMT                             ;OTHERWISE AN ERROR
052310  012777  052344  177206  ESR29:  MOV     #ESR31,@DEVECT          ;ALTER CPU INTR VECTOR
052316  106427  000300                  MTPS    #300
052322  012777  000013  177204          MOV     #13,@CSR2               ;SETS GO, CLRS DONE
052330  000240                          NOP
052332  012777  000000  177174          MOV     #0,@CSR2
052340  000240                          NOP
052342  000401                          BR      ESR30                   ;INTR SHOULD NOT BE HONORED
052344  104000                  ESR31:  EMT                             ;BUT IF HERE, IT WAS

                                ;*******************************************************
                                ;       GENERATE AN INTERRUPT AT LEVEL SEVEN
                                ;*******************************************************
052346  012777  052404  177150  ESR30:  MOV     #ESR33,@DEVECT
052354  012777  000001  177150          MOV     #1,@CSR1                ;CPU AT LEVEL 6, Q22 AT 7
052362  012777  000033  177144          MOV     #33,@CSR2
052370  000240                          NOP
052372  012777  000032  177134          MOV     #32,@CSR2
052400  000240                          NOP
052402  104000                          EMT
052404  012777  052460  177112  ESR33:  MOV     #ESR35,@DEVECT          ;MODIFY INTR VECTOR IN CPU
052412  106427  000340                  MTPS    #340
052416  012777  000033  177110          MOV     #33,@CSR2
052424  000240                          NOP
052426  012777  000032  177100          MOV     #32,@CSR2
052434  000240                          NOP
052436  012777  000001  177070          MOV     #1,@CSR2                ;THIS WILL REMOVE THE
052444  012777  000000  177062          MOV     #0,@CSR2                ;PENDING INTERRUPT
052452  106427  000004                  MTPS    #4                      ;RESTORE THE PSW
052456  000401                          BR      Q22TS1                  ;NADA SHOULD INTERRUPT
052460  104000                  ESR35:  EMT                             ;ERROR IF IT DOES
```

```
                                          ;*****************************************************************************
                                          ;TEST 425          DMA  DATO TRANSFER
                                          ;*****************************************************************************
052462                                    TS425:

                                          ;**************************************************************
                                          ;
                                          ;          THIS TEST WILL SET UP AN ACTUAL DMA TRANSFER
                                          ;
                                          ;**************************************************************

052462  012777  065432  177052            Q22TS1: MOV     #65432,@DATA
052470  012777  001601  177034                    MOV     #1601,@CSR1
052476  005077  177032                            CLR     @CSR2                     ;NO INTERRUPT BITS
052502  012777  177776  177030                    MOV     #177776,@WC                        ;TWO WORDS
052510  012777  050534  177020                    MOV     #CNTR,@BA
052516  005067  176012                            CLR     CNTR
052522  005067  176010                            CLR     CNTR+2                             ;CLEAR THESE FIRST
052526  012777  000001  177014                    MOV     #1,@SIMGOA                         ;SIMULT. GO BIT
052534  032777  000200  176772            ESR26:  BIT     #BIT7,@CSR2               ;WAIT FOR DONE
052542  001001                                    BNE     ESR25
052544  000773                                    BR      ESR26
052546  022767  065432  175760            ESR25:  CMP     #65432,CNTR
052554  001004                                    BNE     ESR32
052556  022767  065432  175752                    CMP     #65432,CNTR+2                      ;GOOD DATA
052564  001401                                    BEQ     Q22TS2
052566  104000                            ESR32:  EMT                               ;NO, BAD DATA



                                          ;*****************************************************************************
                                          ;TEST 426          TEST PWR OK LOGIC
                                          ;*****************************************************************************
052570                                    TS426:

                                          ;**************************************************************
                                          ;
                                          ;          THIS TEST WILL DETERMINE IF THE CPU WILL RESPOND
                                          ;          TO THE BPOK LINE BEING ALTERED. THE Q22BE
                                          ;          WILL BE USED TO CONTROL THAT LINE ON THE Q BUS.
                                          ;
                                          ;**************************************************************

052570  000244                            Q22TS2: CLZ
052572  032777  000400  146604                    BIT     #BIT8,@SWR                ;IS A Q BUS EXCER THERE
052600  001002                                    BNE     ESR20                     ;NO, IT'S A Q22
052602  000167  000460                            JMP     BDVTST                    ;YES, THEN BYPASS THESE TESTS

052606  016767  125212  175712            ESR20:  MOV     24,TMP
052614  016767  125206  175706                    MOV     26,TMP1                   ;SAVE FOR LATER
052622  012767  052674  125174                    MOV     #ESR12,24
052630  012767  000340  125170                    MOV     #340,26           ;SET UP NEW VECTOR
052636  010667  175672                            MOV     SP,CNTR                   ;SAVE STACK POINTER
052642  012706  000420                            MOV     #420,SP                   ;MOVE SP WAY DOWN
052646  012767  052706  125130                    MOV     #ESR13,4                  ;WHERE A STK OVFL WLD GO
```

```
052654 012767 000340 125124            MOV     #340,6
052662 012777 000040 176644            MOV     #40,@CSR2                ;SET BIT 5 IN Q22BE CSR2
052670 000240                          NOP                              ;WILL PULL BPOK H LOW
052672 104000                          EMT                              ;DIDNT CAUSE CPU TO GO
052674 012777 000000 176632    ESR12:  MOV     #0,@CSR2                 ;THRU LOC 24
052702 000240                          NOP                              ;BPOK H WILL GO HIGH
052704 000401                          BR      ESR14
052706 104000                  ESR13:  EMT                              ;IF HERE, AN ERROR BY GOING
                                                                                ;THRU LOC 4
052710 016706 175620           ESR14:  MOV     CNTR,SP          ;RESTORE SOME STUFF
052714 016767 175606 125102            MOV     TMP,24
052722 016767 175602 125076            MOV     TMP1,26
052730 000167 000000                   JMP     Q22TS3                   ;GO TO NEXT TEST



                               ;*************************************************************************
                               ;TEST 427        TEST INDIVIDUAL EXTENDED ADRS BITS
                               ;*************************************************************************
052734                         TS427:

                               ;*******************************************************
                               ;
                               ;       THIS TEST WILL UTILIZE THE Q22BE LATENCY CNTR
                               ;       TO CAPTURE THE EXTENDED ADDRESS BITS ON THE Q BUS.
                               ;       THE Q22BE LATENCY COUNTER BITS 15-12 CORRESPOND
                               ;       TO ADDRESS BITS 21-18.
                               ;
                               ;*******************************************************

052734 016767 125044 175564    Q22TS3: MOV     4,TMP
052742 016767 125040 175560            MOV     6,TMP1                   ;STORE FOR LATER
052750 012767 053046 125026            MOV     #ESR15,4
052756 012767 000340 125022            MOV     #340,6                   ;NEW VECTOR AND PSW

052764 017700 176554                   MOV     @LATCNT,R0                       ;READ IT TO CLR IT
052770 012767 000020 117520            MOV     #BIT4,SR3                        ; 22 BIT ADRSNG
052776 012767 010000 117344            MOV     #10000,KIPAR4                    ;SET FOR ADRS BIT 18
053004 012767 010000 175524            MOV     #10000,CNTR+2
053012 012767 000000 175514            MOV     #0,CNTR                  ;ZERO THESE
053020 012767 000001 124544            MOV     #BIT0,SR0                        ;TURN ON MEM MANG
053026 026727 175502 000017    ESR17:  CMP     CNTR,#17                 ;FINISHED ?
053034 001433                          BEQ     ESR16
053036 105037 100000                   CLRB    @#100000                 ;IF THERE IS NO MEMORY HERE
053042 000240                          NOP                              ;SHOULD HAVE TIMED OUT
053044 000240                          NOP                              ;WE ARE IN BIG TROUBLE

053046 017700 176472           ESR15:  MOV     @LATCNT,R0               ;READ LATENCY COUNTER
053052 042700 007777                   BIC     #7777,R0                 ;CLR DONT CARES
053056 026700 175454                   CMP     CNTR+2,R0                ;EXPECTED ?
053062 001407                          BEQ     ESR18                    ;EQUALS RECVD
053064 012767 000000 124500            MOV     #0,SR0                   ;TURN OFF MM AND 22 BITS
053072 012767 000000 117416            MOV     #0,SR3
053100 104000                          EMT                              ;DATA NOT GOOD
053102 005267 175426           ESR18:  INC     CNTR
053106 062767 010000 117234            ADD     #10000,KIPAR4
```

```
053114  062767  010000  175414                  ADD     #10000,CNTR+2           ;INCREASE ADRS BITS BY ONE
053122  000741                                  BR      ESR17                   ;CONTINUE

053124  016767  175376  124652          ESR16:  MOV     TMP,4
053132  016767  175372  124646                  MOV     TMP1,6                  ;RESTORE THIS VECTOR
053140  012767  000000  124424                  MOV     #0,SR0                  ;TURN OFF MM AND 22 BITS
053146  012767  000000  117342                  MOV     #0,SR3
053154  000167  000106                          JMP     BDVTST



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

053160  012737  000011  001002          ERRORB: MOV     #11,@#$FATAL
053166  012767  000001  125604                  MOV     #1,$MSGTY
053174  032737  000001  001020                  BIT     #1,@#$ENV
053202  001004                                  BNE     Q22HLT
053204  012700  053216                          MOV     #Q22MSG,R0
053210  004767  060114                          JSR     PC,TYPE
053214  000777                          Q22HLT: BR      .
053216  040506  046111  051125          Q22MSG: .ASCIZ  /FAILURE DURING Q BUS EXERCISER TESTS/<12><15>
053224  020105  052504  044522
053232  043516  050440  041040
053240  051525  042440  042530
053246  041522  051511  051105
053254  052040  051505  051524
053262  006412     000
        053266                          .EVEN

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




                                        ;*************************************************************
                                        ;
                                        ;       BDV TESTS
                                        ;
                                        ;
                                        ;*************************************************************

                                        ;*************************************************************************************
                                        ;TEST 430       TEST FOR FUNCTIONALITY OF R/W REGISTER
                                        ;*************************************************************************************
053266                                  TS430:



053266  012767  000017  124230          BDVTST: MOV     #17,LSREG               ;TURN OFF THE 4 LEDS
053274  012737  000012  001004                  MOV     #12,@#$TESTN            ;TST NUMBER FOR APT
053302  012737  054532  000030                  MOV     #ERRORD,@#30            ;ERROR TRAP
053310  000244                                  CLZ
053312  032777  002000  146064                  BIT     #BIT10,@SWR             ;WANT TO TEST E102 ?
053320  001410                                  BEQ     ESR10                   ;NO
053322  016767  124176  175176                  MOV     LSREG,TMP               ;RD THIS TO GET SWITCHES
053330  026727  175172  000113                  CMP     TMP,#113                ;SHOULD BE THIS VALUE
053336  001401                                  BEQ     ESR10                   ;EQUAL
053340  104000                                  EMT
```

```
053342  012737  054436  000030  ESR10:  MOV     #ERRORC,@#30            ;ERROR TRAP
053350  005067  124146                  CLR     RWREG           ;THE NEXT FEW INSTRUCTIONS
053354  016701  124142                  MOV     RWREG,R1                ;WILL TEST THE R/W ABILITY
053360  020127  000000                  CMP     R1,#0                   ;OF THE R/W MAINT. REGISTER
053364  001401                          BEQ     ESR40
053366  104000                          EMT
053370  012767  177777  124124  ESR40:  MOV     #-1,RWREG
053376  016701  124120                  MOV     RWREG,R1
053402  022701  177777                  CMP     #177777,R1
053406  001401                          BEQ     ESR41
053410  104000                          EMT
053412  012767  125252  124102  ESR41:  MOV     #125252,RWREG
053420  016701  124076                  MOV     RWREG,R1
053424  020127  125252                  CMP     R1,#125252
053430  001401                          BEQ     ESR42
053432  104000                          EMT
053434  105067  124062          ESR42:  CLRB    RWREG                   ;BYTES
053440  016701  124056                  MOV     RWREG,R1
053444  020127  125000                  CMP     R1,#125000
053450  001401                          BEQ     ESR43
053452  104000                          EMT
053454  000367  124042          ESR43:  SWAB    RWREG
053460  016701  124036                  MOV     RWREG,R1
053464  020127  000252                  CMP     R1,#252
053470  001401                          BEQ     ESR44
053472  104000                          EMT
053474  012767  052525  124020  ESR44:  MOV     #52525,RWREG
053502  016701  124014                  MOV     RWREG,R1
053506  020127  052525                  CMP     R1,#52525
053512  001401                          BEQ     ESR45
053514  104000                          EMT
053516                          ESR45:
053516  105067  124001                  CLRB    RWREG+1
053522  016701  123774                  MOV     RWREG,R1
053526  020127  000125                  CMP     R1,#125
053532  001401                          BEQ     ESR46
053534  104000                          EMT
053536  000367  123760          ESR46:  SWAB    RWREG
053542  016701  123754                  MOV     RWREG,R1
053546  020127  052400                  CMP     R1,#52400
053552  001401                          BEQ     ESR47
053554  104000                          EMT
053556  005067  123740          ESR47:  CLR     RWREG
053562  052767  100000  123732          BIS     #BIT15,RWREG            ;SET 15
053570  016701  123726                  MOV     RWREG,R1
053574  020167  123722          ROTLP1: CMP     R1,RWREG                ;ARE THEY THE SAME
053600  001401                          BEQ     ESR48
053602  104000                          EMT
053604  006001                  ESR48:  ROR     R1                      ;NEXT BIT TO THE RIGHT
053606  001403                          BEQ     ESR49
053610  006067  123706                  ROR     RWREG
053614  000767                          BR      ROTLP1                  ;CONTINUE TESTING
053616  012767  177777  123676  ESR49:  MOV     #-1,RWREG
053624  042767  100000  123670          BIC     #BIT15,RWREG
053632  016701  123664                  MOV     RWREG,R1
053636  026701  123660          ROTLP2: CMP     RWREG,R1                ;SAME ?
```

```
053642  001401                          BEQ     ESR50                   ;YES
053644  104000                          EMT
053646  000261                  ESR50:  SEC                             ;SET THE C BIT
053650  006067  123646                  ROR     RWREG
053654  006001                          ROR     R1
053656  020127  077777                  CMP     R1,#77777
053662  001365                          BNE     ROTLP2                  ;NOT FINISHED YET
053664  000461                          BR      BDVTS2


                                ;*************************************************************************
                                ;TEST 431       ROM CHECKSUM TEST
                                ;*************************************************************************
053666                          TS431:

                                ;************************************************************
                                ;
                                ;       THE PREVIOUS TEST CHECKED OUT THE ROM R/W REGISTER
                                ;       NOW WE WILL TEST THE 2K DIAGNOSTIC ROM FOR THE
                                ;       CORRECT CHECKSUM AND CHECKWORD
                                ;
                                ;
                                ;************************************************************



                                    ;THE CHECKWORDS CORRESPONDING TO ROM CHIPS #23-045E2 AND #23-046E2 FOLLOW:
053666                          SFPTBL:
053666  017042                          .WORD   17042           ;ROMA: PAGE 0,1
053670  020656                          .WORD   20656           ;ROMB: PAGE 2,3
053672  065162                          .WORD   65162           ;ROMC: PAGE 4,5
053674  161744                          .WORD   161744          ;ROMD: PAGE 6,7
053676  124453                          .WORD   124453          ;ROME: PAGE 10,11
053700  113667                          .WORD   113667          ;ROMF: PAGE 12,13
053702  056040                          .WORD   56040           ;ROMG: PAGE 14,15
053704  044734                          .WORD   44734           ;ROMH: PAGE 16,17

                                    ;THE CHECKWORDS CORRESPONDING TO ROM CHIPS #23-339E2 AND #23-340E2 FOLLOW:
053706  166020                          .WORD   166020          ;ROMA: PAGE 0,1
053710  020232                          .WORD   20232           ;ROMB: PAGE 2,3
053712  045651                          .WORD   45651           ;ROMC: PAGE 4,5
053714  036474                          .WORD   36474           ;ROMD: PAGE 6,7
053716  066675                          .WORD   66675           ;ROME: PAGE 10,11
053720  163100                          .WORD   163100          ;ROMF: PAGE 12,13
053722  005407                          .WORD   5407            ;ROMG: PAGE 14,15
053724  022243                          .WORD   22243           ;ROMH: PAGE 16,17

                                    ;THE CHECKWORDS CORRESPONDING TO ROM CHIPS #23-010E2 AND #23-011E2 FOLLOW:
053726  031547                          .WORD   31547           ;ROMA: PAGE 0,1
053730  014036                          .WORD   14036           ;ROMB: PAGE 2,3
053732  065162                          .WORD   65162           ;ROMC: PAGE 4,5
053734  124632                          .WORD   124632          ;ROMD: PAGE 6,7
053736  032040                          .WORD   32040           ;ROME: PAGE 10,11
053740  167124                          .WORD   167124          ;ROMF: PAGE 12,13
053742  155461                          .WORD   155461          ;ROMG: PAGE 14,15
053744  032257                          .WORD   32257           ;ROMH: PAGE 16,17

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

```
                                          ;
                                          ;       DATA SECTION FOR THE NEXT TEST
                                          ;
                                          ;***************************************************************
053746  000000                                    VRTPCR: .WORD   0               ;VIRTUAL PAGE CONTROL REGISTER
053750  000000                                    BCF:    .WORD   0
053752  000000                                    COUNTR: .WORD   0
053754  000001                                    ANSR:   .WORD   1
053756  000000                                    RFLAG:  .WORD   0
053760  000000                                    EXPSUM: .WORD   0
053762  000000                                    ACTSUM: .WORD   0

                                          ;************************************************************
                                          ;FUNCTIONAL DESCRIPTION:
                                          ;SUBROUTINE TO COMPUTE A CHECKSUM IN A ROM/EPROM
                                          ;INPUT: CONTENTS OF BCF
                                          ;IMPLICIT INPUTS: CONTENTS OF PCR
                                          ;OUTPUT: A CHECKSUM VALUE STORED IN LOCATION ACTSUM
                                          ;CALLING SEQUENCE: JSR PC,CHKSUM
                                          ;************************************************************

053764  012701  173776                    CHKSUM: MOV     #173776,R1      ;STORE THE HIGHEST ADDRESS IN THE ROM
053770  066701  177754                            ADD     BCF,R1          ;FOR EITHER LOW OR HIGH BYTES
053774  005067  177762                            CLR     ACTSUM          ;CLEAR LOCATION WHICH WILL HOLD THE CHECKSUM
054000  012702  173000                            MOV     #173000,R2      ;COMPUTE THE LOWEST ADDRESS IN THE ROM
054004  066702  177740                            ADD     BCF,R2          ;WHERE THE DATA WILL START
054010  111204                            1$:     MOVB    (R2),R4         ;GET DATA IN BYTES
054012  060467  177744                            ADD     R4,ACTSUM       ;ADD CONTENTS OF EACH LOCATION TO THE CHECKSUM
054016  062702  000002                            ADD     #2,R2           ;ADJUST ADDRESS
054022  020201                                    CMP     R2,R1           ;COMPARE CURRENT ADDRESS WITH HIGHEST ADDRESS
054024  002771                                    BLT     1$              ;BR IF LESS THAN
054026  000207                                    RTS     PC              ;RETURN



                                          ;******************************************** ******************
                                          ;TEST TO PERFORM CHECKSUM AND CHECKWORD VERIFICATION ON THE 2K
                                          ;OF DIAGNOSTIC ROM. IN UNATTENDED MODE, THE ROM WILL BE ADDRESSED
                                          ;FROM 0-2K. IN STAND-ALONE MODE, THE OPERATOR MAY CHANGE THE
                                          ;ADDRESS BY RESPONDING TO QUESTIONS GENERATED ON THE FIRST PASS.
                                          ;***************************************************************

054030  012767  000400  000352            BDVTS2: MOV     #400,DRLP       ;STORE STARTING ADDRESS
054036  016767  000346  177702                    MOV     DRLP,VRTPCR     ;SET UP PCR
054044  016767  177676  123446                    MOV     VRTPCR,PCR
054052  012767  000010  177672                    MOV     #10,COUNTR      ;SET NUMBER OF CHECKWORDS TO CHECK
054060  012705  053666                            MOV     #SFPTBL,R5      ;LOCATION OF CHECKWORDS
054064  012767  000001  177664                    MOV     #1,RFLAG        ;INDICATE ROM
054072  005067  177652                    DLOOP:  CLR     BCF             ;SIGNAL LOW BYTES ARE BEING CHECKED
054076  122737  177777  173774                    CMPB    #-1,@#173774    ;DOES THE ROM EXIST?
054104  001001                                    BNE     1$              ;BR IF YES
054106  104000                                    EMT
054110  004767  177650                    1$:     JSR     PC,CHKSUM       ;COMPUTE THE ACTUAL CHECKSUM
054114  113767  173776  177636                    MOVB    @#173776,EXPSUM ;GET THE STORED CHECKSUM
054122  066767  177634  177630                    ADD     ACTSUM,EXPSUM   ;ADD THE EXPECTED AND ACTUAL CHECKSUMS
054130  105767  177624                            TSTB    EXPSUM          ;BYTE RESULT = 0?
054134  001401                                    BEQ     2$              ;BR IF YES
```

```
054136  104000                           EMT
054140  012767  000001  177602   2$:     MOV     #1,BCF                   ;SET BCF TO DENOTE HIGH BYTES
054146  122737  177777  173775           CMPB    #-1,@#173775             ;DOES THE ROM EXIST?
054154  001001                           BNE     3$                       ;BR IF YES
054156  104000                           EMT
054160  004767  177600           3$:     JSR     PC,CHKSUM                ;COMPUTE THE ACTUAL CHECKSUM
054164  113767  173777  177566           MOVB    @#173777,EXPSUM          ;GET EXPECTED CHECKSUM
054172  066767  177564  177560           ADD     ACTSUM,EXPSUM            ;ADD THE EXPECTED AND ACTUAL CHECKSUMS
054200  105767  177554                   TSTB    EXPSUM                   ;BYTE RESULT = 0?
054204  001401                           BEQ     4$                       ;BR IF YES
054206  104000                           EMT
054210  062767  001002  177530   4$:     ADD     #1002,VRTPCR             ;NEXT PAGE IN PCR
054216  016767  177524  123274           MOV     VRTPCR,PCR
054224  005367  177522                   DEC     COUNTR                   ;DECREMENT CHECKWORD COUNT
054230  001320                           BNE     DLOOP                    ;LOOP UNTIL ALL 20 PAGES HAVE BEEN CHECKED


                                 ;GET THE CHECKWORDS FROM THE ROMS AND PUT INTO TABLE 'CHKWRD'
054232  012702  054412                   MOV     #CHKWRD,R2
054236  012767  000001  177502           MOV     #1,VRTPCR
054244  012767  000010  177500           MOV     #10,COUNTR
054252  016767  177470  123240   5$:     MOV     VRTPCR,PCR
054260  013722  173376                   MOV     @#173376,(R2)+
054264  062767  000002  177454           ADD     #2,VRTPCR
054272  005367  177454                   DEC     COUNTR
054276  001365                           BNE     5$


                                 ;TRY TO IDENTIFY THE ROM CHIPS
054300  016701  000126                   MOV     TABLES,R1
054304  012767  053666  000122           MOV     #SFPTBL,PNTR
054312  016700  000116           SIXDLR: MOV     PNTR,R0
054316  012702  054412                   MOV     #CHKWRD,R2
054322  012767  000010  177422           MOV     #10,COUNTR
054330  022022                   7$:     CMP     (R0)+,(R2)+              ;ARE THE CHECKWORDS EQUAL?
054332  001017                           BNE     NOTEQ                    ;BRANCH IF NOT
054334  005367  177412                   DEC     COUNTR                   ;DONE CHECKING THIS TABLE?
054340  001373                           BNE     7$                       ;BRANCH IF NOT
054342  020127  000003                   CMP     R1,#3            ;DID THE FIRST TABLE OF CHECKWORDS COMPARE?
054346  001001                           BNE     8$               ;BRANCH IF NOT
054350  000416                           BR      COMPLE
054352  020127  000002           8$:     CMP     R1,#2            ;DID THE SECOND TABLE OF CHECKWORDS COMPARE?
054356  001001                           BNE     NOCMP            ;BRANCH IF NOT
054360  000412                           BR      COMPLE
054362  020127  000001           NOCMP:  CMP     R1,#1
054366  001001                           BNE     NOTEQ
054370  000406                           BR      COMPLE
054372  062767  000020  000034   NOTEQ:  ADD     #20,PNTR
054400  005301                           DEC     R1                       ;ANY MORE TABLES TO CHECK?
054402  001343                           BNE     SIXDLR                   ;BRANCH IF YES
054404  104000                           EMT
054406  000515                   COMPLE: BR      FPSTRT
```

```
054410 000000                          DRLP:   .WORD   0


054412 000010                          CHKWRD: .BLKW   10              ;TABLE TO STORE THE CHECKWORDS

054432 000003                          TABLES: .WORD   3               ;NUMBER OF CHECKWORD TABLES
054434 000000                          PNTR:   .WORD   0               ;WILL BE USED AS A POINTER

                                       ;*************************************************************
                                       ;
                                       ;       ERROR ROUTINE FOR THE BDV TESTING
                                       ;
                                       ;*************************************************************

054436 012737 000012 001002            ERRORC: MOV     #12,@#$FATAL
054444 012767 000001 124326                    MOV     #1,$MSGTY
054452 032737 000001 001020                    BIT     #1,@#$ENV                       ;UNDER APT ?
054460 001004                                  BNE     BDVHLT
054462 012700 054474                           MOV     #BDVMSG,R0
054466 004767 056636                           JSR     PC,TYPE
054472 000777                          BDVHLT: BR      .

054474 040506 046111 042105            BDVMSG: .ASCIZ  /FAILED DURING THE BDV TESTS/<12><15>
054502 042040 051125 047111
054510 020107 044124 020105
054516 042102 020126 042524
054524 052123 005123 000015


054532 012737 000012 001002            ERRORD: MOV     #12,@#$FATAL
054540 012767 000001 124232                    MOV     #1,$MSGTY
054546 032737 000001 001020                    BIT     #1,@#$ENV
054554 001346                                  BNE     BDVHLT
054556 012700 054570                           MOV     #SWMSG,R0
054562 004767 056542                           JSR     PC,TYPE
054566 000000                                  HALT
054570 044103 041505 020113            SWMSG:  .ASCIZ  /CHECK SWITCHES ON E102, RESTART AT 200/<12><15>
054576 053523 052111 044103
054604 051505 047440 020116
054612 030505 031060 020054
054620 042522 052123 051101
054626 020124 052101 031040
054634 030060 006412    000
       054642                          .EVEN

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









       000244                          FPVECT=244
                                               .SBTTL  FPP REGISTER DEFINITIONS
```

```
          000000                          AC0     =%0
          000001                          AC1     =%1
          000002                          AC2     =%2
          000003                          AC3     =%3
          000004                          AC4     =%4
          000005                          AC5     =%5
          000006                          AC6     =%6
          000007                          AC7     =%7


054642    012706  001000                  FPSTRT: MOV     #STBOT,SP          ;SET UP STACK POINTER
054646    000244                                  CLZ
054650    032777  000002  144526                  BIT     #2,@SWR
054656    001002                                  BNE     1$
054660    000167  047772                          JMP     SLU1ST
054664    012737  124476  000030          1$:     MOV     #ERROR4,@#30       ;SETUP FOR CORRECT ERROR CALL
054672    012737  000003  001004                  MOV     #3,@#$TESTN        ;PUT TEST NUMBER IN MAILBOX


                                          ;*****************************************************************************
                                          ;TEST 432          LDFPS, STFPS AND DATA PATHS TEST
                                          ;*****************************************************************************
054700                                    TS432:
054700    012700  177777                          MOV     #-1,R0                       ;INITIALIZE THE COUNT PATTERN.
054704    012737  054756  000244                  MOV     #AERR1,@#FPVECT              ;SET UP FOR UNABLE TO DECODE
054712    012737  054756  000010                  MOV     #AERR1,@#10                  ;FPP INSTRUCTION TRAP TO 244 OR 10.
054720    012737  054756  000004                  MOV     #AERR1,@#ERRVECT             ;IF EITHER INSTRUCTION
                                                                                       ;FAILS TO GO THROUGH THE
                                                                                       ;CORRECT SRC OR DST MODE AN
                                                                                       ;ODD ADDRESS TRAP WILL OCCUR.
054726                                    A1:
054726    010004                          A11:    MOV     R0,R4
054730    042704  030020                          BIC     #30020,R4
054734    170104                                  LDFPS   R4                           ;TEST INSTRUCTION.

054736    012701  177777                          MOV     #-1,R1
054742    170201                          A12:    STFPS   R1                           ;TEST INSTRUCTION.
054744    010004                                  MOV     R0,R4                        ;MASK OFF UNSETTABLE BITS.
054746    042704  030020                          BIC     #30020,R4
054752    020401                                  CMP     R4,R1                        ;COMPARE DATA EXPECTED WITH
                                                                                       ;THE DATA READ.
054754    001401                                  BEQ     A2
054756                                    AERR1:
054756    104000                                  EMT                        ;

054760    012700  000001                  A2:     MOV     #1,R0                        ;NEXT PATTERN WILL BE ALL ZERO
054764    077020                                  SOB     R0,A1                        ;DECREMENT COUNT PATTERN
054766                                    ADONE:
054766    004767  047600                          JSR     PC,.RSET           ;GO INITIALIZE THE FPS AND STACK; AND
                                                                             ;SEE IF THE USER HAS EXPRESSED
                                                                             ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                             ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                             ;THE USER TYPED CONTROL G?).


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

```
                                   ;TEST 433          CFCC TEST
                                   ;*****************************************************************************
054772                             TS433:
054772  012700  000017                       MOV     #17,R0                          ;R0 CONTAINS TO TEST PATTERN.
054776                             B1:
054776  170100                               LDFPS   R0                              ;LOAD THE TEST PATTERN
055000                             B2:
055000  170000                               CFCC                                    ;COPY CONDITION CODES.
055002  013703  177776                       MOV     @#PSW,R3                        ;SEE IF PATTERN TRANSFERED.
055006  042703  177760                       BIC     #177760,R3
055012  020003                               CMP     R0,R3
055014  001401                               BEQ     B3
055016  104000                               EMT                     ;
055020  077012                     B3:       SOB     R0,B1
055022                             BDONE:
055022  004767  047544                       JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                         ;SEE IF THE USER HAS EXPRESSED
                                                                         ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                         ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                         ;THE USER TYPED CONTROL G?).

                                   ;*****************************************************************************
                                   ;TEST 434          SETF, SETD, SETI AND SETL TEST
                                   ;*****************************************************************************
055026                             TS434:
055026  005000                               CLR     R0
055030  170100                               LDFPS   R0                              ;CLEAR THE FPS.
055032  170001                     C15:      SETF                                    ;TEST INSTRUCTION.
055034  170201                               STFPS   R1                              ;GET RESULT.
055036  005002                               CLR     R2
055040  020201                               CMP     R2,R1                           ;DID AN ERROR OCCUR?
055042  001401                               BEQ     C2
055044  104000                               EMT                     ;
055046  012700  147757             C2:       MOV     #147757,R0
055052  170100                               LDFPS   R0                              ;PUT 147757 IS FPS
055054  170001                     C25:      SETF                                    ;CLEAR FD BIT.
055056  170201                               STFPS   R1                              ;GET RESULT
055060  012702  147557                       MOV     #147557,R2
055064  020102                               CMP     R1,R2                           ;RESULT CORRECT.
055066  001401                               BEQ     C3
055070  104000                               EMT                     ;
055072  012700  147757             C3:       MOV     #147757,R0
055076  170100                               LDFPS   R0                              ;LOAD 147757 INTO FPS.
055100  170011                     C35:      SETD                                    ;SETD FD BIT.
055102  170201                               STFPS   R1
055104  012702  147757                       MOV     #147757,R2
055110  020102                               CMP     R1,R2                           ;RESULT CORRECT?
```

```
055112  001401                   BEQ     C4
055114  104000                   EMT                     ;
055116  005000          C4:      CLR     R0
055120  170100                   LDFPS   R0                          ;CLEAR FPS.
055122  170011          C45:     SETD                                ;SET FD BIT.

055124  170201                   STFPS   R1                          ;GET RESULT.
055126  012702  000200           MOV     #200,R2
055132  020102                   CMP     R1,R2                       ;RESULT CORRECT?
055134  001401                   BEQ     C5
055136  104000                   EMT                     ;
055140  005000          C5:      CLR     R0

055142  170100                   LDFPS   R0                          ;CLEAR FPS
055144  170002          C55:     SETI                                ;CLEAR FL BIT.

055146  170201                   STFPS   R1                          ;GET RESULT.
055150  005002                   CLR     R2
055152  020201                   CMP     R2,R1                       ;RESULT CORRECT?
055154  001401                   BEQ     C6
055156  104000                   EMT                     ;
055160  012700  147757  C6:      MOV     #147757,R0
055164  170100                   LDFPS   R0                          ;PUT 147757 INTO FPS
055166  170002          C65:     SETI                                ;CLEAR FL BIT.

055170  170201                   STFPS   R1                          ;GET THE RESULT.
055172  012702  147657           MOV     #147657,R2
055176  020102                   CMP     R1,R2                       ;RESULT CORRECT?
055200  001401                   BEQ     C7
055202  104000                   EMT                     ;
055204  012700  147757  C7:      MOV     #147757,R0
055210  170100                   LDFPS   R0                          ;SET FPS TO 147757.
055212  170012          C75:     SETL                                ;SET FL BIT.

055214  170201                   STFPS   R1                          ;GET THE RESULT.
055216  012702  147757           MOV     #147757,R2
055222  020102                   CMP     R1,R2                       ;RESULT CORRECT?
055224  001401                   BEQ     C8
055226  104000                   EMT                     ;
055230  005000          C8:      CLR     R0
055232  170100                   LDFPS   R0                          ;CLEAR FPS.
055234  170012          C85:     SETL                                ;SET FL BIT.

055236  170201                   STFPS   R1
055240  012702  000100           MOV     #100,R2
055244  020102                   CMP     R1,R2                       ;RESULT CORRECT.
055246  001401                   BEQ     CDONE
055250  104000                   EMT                     ;
055252                  CDONE:
055252  004767  047314           JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                         ;SEE IF THE USER HAS EXPRESSED
                                                         ;THE DESIRE TO CHANGE THE SOFTWARE
                                                         ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                         ;THE USER TYPED CONTROL G?).
```

```
                                        ;*************************************************************************
                                        ;TEST 435        ILLEGAL FPP OP CODES AND STST TEST
                                        ;*************************************************************************
055256                                  TS435:
055256  012705  170003                          MOV     #170003,R5                      ;INITIAL OP CODE.
055262  012737  055322  000004                  MOV     #DERR2,@#ERRVECT
055270  012737  055354  000244                  MOV     #DERR1,@#FPVECT

055276  005000                          D1:     CLR     R0
055300  170100                                  LDFPS   R0                              ;CLEAR FPS.
055302  005002                                  CLR     R2
055304  010537  055310                          MOV     R5,@#D2                         ;SET UP THE ILLEGAL INSTRUCTION.
055310  000000                          D2:     .WORD   0
055312  170000                          D3:     CFCC
055314  005202                                  INC     R2
055316  005202                          D4:     INC     R2

055320  170201                                  STFPS   R1                              ;REPORT FAILURE. DID NOT TRAP.
055322                                  DERR2:
055322  104000                                  EMT                     ;
055324  022705  170010                  D5:     CMP     #170010,R5                      ;COMPUTE NEXT OP CODE
055330  001003                                  BNE     D6
055332  012705  170013                          MOV     #170013,R5
055336  000757                                  BR      D1

055340  022705  170077                  D6:     CMP     #170077,R5
055344  001001                                  BNE     D7
055346  000424                                  BR      DDONE
055350  005205                          D7:     INC     R5
055352  000751                                  BR      D1

055354  022716  055312                  DERR1:  CMP     #D3,(SP)                        ;DID TRAP OCCUR ON TEST INSTRUCTION?
055360  001401                                  BEQ     1$
055362  104000                                  EMT                     ;
055364  022626                          1$:     CMP     (SP)+,(SP)+
055366  170201                                  STFPS   R1                              ;GET THE FPS AND SEE IF IT IS
055370  022701  100000                          CMP     #100000,R1                      ;SET CORRECTLY.
055374  001401                                  BEQ     3$
055376  104000                                  EMT                     ;
055400  012704  000001                  3$:     MOV     #1,R4
055404  170304                          D8:     STST    R4                              ;GET THE FEC CODE. NOTE THAT
                                                                                        ;IF THE DESTINATION MODE IS
                                                                                        ;IMPROPERLY DECODED AN ODD
                                                                                        ;ADDRESS TRAP TO 4 SHOULD OCCUR.
055406  022704  000002                          CMP     #2,R4                           ;WAS FEC CORRECT?
055412  001001                                  BNE     D9
055414  000743                                  BR      D5

055416                                  D9:                                             ;REPORT STST FAILURE
055416  104000                                  EMT                     ;
055420                                  DDONE:
055420  004767  047146                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).
```

```

                                        ;*******************************************************************************
                                        ;TEST 436        FID, INTERRUPT DISABLE, BIT TEST
                                        ;*******************************************************************************
055424                                  TS436:
055424  012737  055464  000244                  MOV     #EERRO,@#FPVECT         ;SETUP FOR THE INTERRUPT.

055432  012700  040000                  E1:     MOV     #40000,R0
055436  170100                                  LDFPS   R0                      ;SET FID.
055440  170020                          E3:     .WORD   170020                  ;ILLEGAL FPP INSTRUCTION.
055442  170000                          E4:     CFCC

055444  170201                                  STFPS   R1                      ;SEE IF ERROR WAS DETECTED.
055446  022701  140000                          CMP     #140000,R1
055452  001004                                  BNE     EERRO

055454  170304                                  STST    R4                      ;SEE IF FEC=2
055456  022704  000002                          CMP     #2,R4
055462  001401                                  BEQ     EDONE
055464                                  EERRO:
055464  104000                                  EMT                     ;
055466                                  EDONE:
055466  004767  047100                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).


                                        ;*******************************************************************************
                                        ;TEST 437        LDD AND STD, WITH SRC AND DST MODE 1, TEST
                                        ;*******************************************************************************
055472                                  TS437:

055472  005000                                  CLR     R0
055474  170100                                  LDFPS   R0
055476  170011                                  SETD
055500  012701  055742                          MOV     #FDATI0,R1              ;SET UP THE LOAD DATA.
055504  012702  056006                          MOV     #FXDAT0,R2
055510  012703  000010                          MOV     #10,R3

055514  012221                          F2:     MOV     (R2)+,(R1)+
055516  077302                                  SOB     R3,F2

055520  012700  055752                          MOV     #FDATI4,R0              ;SETUP R0 FOR THE LDD (R0),AC0.
055524  012737  055740  000004                  MOV     #FERR20,@#ERRVECT       ;IF THE SRC FLOWS FAIL THEN
                                                                                ;AN ODD ADDRESS MAY OCCUR.
055532  005003                                  CLR     R3

055534  172410                          F3:     LDD     (R0),AC0
055536  005203                          F4:     INC     R3
055540  005203                                  INC     R3

055542  020027  055752                          CMP     R0,#FDATI4              ;WAS R0 AFFECTED?
055546  001401                                  BEQ     F5
055550  104000                                  EMT                     ;
```

```
055552  020327  000002          F5:     CMP     R3,#2                   ;SEE IF THE PC WAS ADVERSELY
055556  001401                          BEQ     1$
055560  104000                          EMT                     ;
055562  012701  055742          1$:     MOV     #FDATI0,R1              ;MAKE SURE THE SOURCE DATA WAS
055566  012702  056006                  MOV     #FXDAT0,R2              ;NOT AFFECTED.
055572  012703  000010                  MOV     #10,R3
055576  022122                  2$:     CMP     (R1)+,(R2)+
055600  001401                          BEQ     3$
055602  104000                          EMT                     ;
055604  077304                  3$:     SOB     R3,2$

055606  170201                          STFPS   R1                      ;MAKE SURE THE FPS IS CORRECT.
055610  022701  000200                  CMP     #200,R1
055614  001401                          BEQ     F6
055616  104000                          EMT                     ;
055620  012703  177777          F6:     MOV     #-1,R3
055624  012704  000010                  MOV     #10,R4
055630  012705  055764                  MOV     #FDAT00,R5              ;SET UP THE OUTPUT DATA BUFFER.
055634  010325                  F7:     MOV     R3,(R5)+
055636  077402                          SOB     R4,F7

055640  012700  055774                  MOV     #FDAT04,R0              ;SET UP R0 FOR DST MODE 1 REG 0.
055644  012737  055740  000004          MOV     #FERR20,@#ERRVECT       ;IF THE DST FLOWS FAIL AN ODD
                                                                        ;ADDRESS COULD OCCUR.
055652  005003                          CLR     R3

055654  174010                  F10:    STD     AC0,(R0)                ;TEST INSTRUCTION.
055656  005203                  F11:    INC     R3
055660  005203                          INC     R3

055662  020027  055774                  CMP     R0,#FDAT04              ;WAS R0 MODIFIED?
055666  001401                          BEQ     F12
055670  104000                          EMT                     ;
055672  020327  000002          F12:    CMP     R3,#2                   ;WAS THE PC AFFECTED CORRECTLY?
055676  001401                          BEQ     F135
055700  104000                          EMT                     ;
055702  012701  055764          F135:   MOV     #FDAT00,R1
055706  012702  056006                  MOV     #FXDAT0,R2
055712  012703  000010                  MOV     #10,R3          ;SETUP LOOP COUNT
055716  022122                  F13:    CMP     (R1)+,(R2)+     ;WAS DATA OUTPUT CORRECTLY
055720  001401                          BEQ     F14
055722  104000                          EMT                     ;
055724  077304                  F14:    SOB     R3,F13          ;SUBTRACT 1 FROM LOOP COUNT AND LOOP IF NOT ZERO
055726  005001                  F22:    CLR     R1
055730  170201                          STFPS   R1                      ;MAKE SURE FPS IS CORRECT.
055732  022701  000200                  CMP     #200,R1
055736  001433                          BEQ     FDONE
055740                          FERR20:
055740  104000                          EMT                     ;

055742  177777                  FDATI0: -1
055744  177777                  FDATI1: -1
055746  177777                  FDATI2: -1
055750  177777                  FDATI3: -1
055752  177777                  FDATI4: -1
055754  177777                  FDATI5: -1
```

```
055756  177777                  FDATI6: -1
055760  177777                  FDATI7: -1
055762  177777                          -1
055764  177777                  FDAT00: -1
055766  177777                  FDAT01: -1
055770  177777                  FDAT02: -1
055772  177777                  FDAT03: -1
055774  177777                  FDAT04: -1
055776  177777                  FDAT05: -1
056000  177777                  FDAT06: -1
056002  177777                  FDAT07: -1
056004  177777                          -1
056006  177777                  FXDAT0: -1
056010  177777                  FXDAT1: -1
056012  177777                  FXDAT2: -1
056014  177777                  FXDAT3: -1
056016  052525                  FXDAT4: 052525
056020  031463                  FXDAT5: 031463
056022  007417                  FXDAT6: 007417
056024  000477                  FXDAT7: 000477


056026                          FDONE:
056026  004767  046540                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).



                                ;*************************************************************************
                                ;TEST 440        FSRC MODE 0 TEST
                                ;*************************************************************************
056032                          TS440:
056032                          I1:
056032  170011                          SETD                            ;SET FD.
056034  012700  056320                  MOV     #IDATI0,R0
056040  012701  056270                  MOV     #IPAT10,R1
056044  012702  000004                  MOV     #4,R2
056050  012120                  I2:     MOV     (R1)+,(R0)+             ;SET UP THE INPUT DATA BUFFER.
056052  077202                          SOB     R2,I2

056054  012700  056320                  MOV     #IDATI0,R0              ;LOAD AC1
056060  172510                          LDD     (R0),AC1

056062  012700  056300                  MOV     #IPAT20,R0              ;LOAD AC0
056066  172410                          LDD     (R0),AC0

056070  012701  000001                  MOV     #1,R1                   ;IN CASE THE FSRC FLOWS FAIL
056074  172401                  I3:     LDD     AC1,AC0                 ;TEST INSTRUCTION.
056076  000240                  I4:     NOP
056100  000240                  I5:     NOP

056102  012700  056310                  MOV     #IDAT00,R0
056106  174010                          STD     AC0,(R0)                ;GET AC0, THE RESULTS.
```

```
056110  012700  056310                  MOV     #IDAT00,R0              ;SEE IF DATA IS CORRECT.
056114  012701  056320                  MOV     #IDATI0,R1
056120  012702  000004                  MOV     #4,R2
056124  022021                  I6:     CMP     (R0)+,(R1)+
056126  001401                          BEQ     I105
056130  104000                          EMT                     ;
056132  077204                  I105:   SOB     R2,I6

                                ;NOW TEST THE LOAD INSTRUCTION WITH FSRC MODE ZERO AND FD CLEAR.

056134  012700  056270          I12:    MOV     #IPAT10,R0
056140  012701  056320                  MOV     #IDATI0,R1
056144  012702  000004                  MOV     #4,R2
056150  012021                  I13:    MOV     (R0)+,(R1)+
056152  077202                          SOB     R2,I13

056154  012700  056320                  MOV     #IDATI0,R0              ;SET UP AC1
056160  172510                          LDD     (R0),AC1

056162  012700  056300                  MOV     #IPAT20,R0              ;SET UP AC0
056166  172410                          LDD     (R0),AC0

056170  012701  000001                  MOV     #1,R1
056174  170001                          SETF                            ;CLEAR FD.

056176  172401                  I14:    LDF     AC1,AC0                 ;TEST INSTRUCTION.
056200  000240                  I15:    NOP
056202  000240                  I16:    NOP

056204  170200                          STFPS   R0                      ;SEE IF FPS IS STILL CLEAR.
056206  022700  000004                  CMP     #4,R0
056212  001401                          BEQ     I17
056214  104000                          EMT                     ;
056216                          I17:                                    ;RESET TO DOUBLE MODE.
056216  170011                          SETD

056220  012700  056310                  MOV     #IDAT00,R0
056224  174010                          STD     AC0,(R0)                ;GET AC0

056226  012737  177777  056324          MOV     #-1,@#IDATI2
056234  012737  177777  056326          MOV     #-1,@#IDATI3
056242  012700  056310                  MOV     #IDAT00,R0
056246  012701  056320                  MOV     #IDATI0,R1
056252  012702  000004                  MOV     #4,R2
056256  022021                  I20:    CMP     (R0)+,(R1)+             ;SEE IF AC0 WAS CORRECT.
056260  001401                          BEQ     I23
056262  104000                          EMT                     ;
056264  077204                  I23:    SOB     R2,I20
056266  000420                          BR      IDONE                   ;NO ERRORS.

056270  000000                  IPAT10: 0
056272  170360                  IPAT11: 170360
056274  016161                  IPAT12: 016161
056276  052525                  IPAT13: 052525

056300  177777                  IPAT20: -1
```

```
056302  177777                          IPAT21: -1
056304  177777                          IPAT22: -1
056306  177777                          IPAT23: -1

056310  000000                          IDAT00: 0
056312  000000                          IDAT01: 0
056314  000000                          IDAT02: 0
056316  000000                          IDAT03: 0

056320  000000                          IDAT10: 0
056322  000000                          IDAT11: 0
056324  000000                          IDAT12: 0
056326  000000                          IDAT13: 0

056330                                  IDONE:
056330  004767  046236                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).


                                        ;*******************************************************************************
                                        ;TEST 441               FDST MODE 0 TEST
                                        ;*******************************************************************************
056334                                  TS441:
056334  170011                                  SETD                                    ;SET FD
056336  012700  056574                          MOV     #TPAT10,R0
056342  012701  056624                          MOV     #TDAT10,R1
```

```
12841  056346  012702  000004          MOV     #4,R2
```

```
12842  056352  012021          T2:     MOV     (R0)+,(R1)+             ;SET UP THE INPUT DATA BUFFER.
12843  056354  077202                  SOB     R2,T2
12844
```

```
056356  012700  056624                   MOV     #TDATI0,R0              ;LOAD AC0
056362  172410                           LDD     (R0),AC0

056364  012700  056604                   MOV     #TPAT20,R0              ;LOAD AC1
056370  172510                           LDD     (R0),AC1

056372  012701  000001                   MOV     #1,R1                   ;IF THE (BUT FDST) FORK FAILS
056376  174001                   T3:     STD     AC0,AC1
056400  000240                   T4:     NOP
056402  000240                   T5:     NOP

056404  012700  056614                   MOV     #TDAT00,R0
056410  174110                           STD     AC1,(R0)                ;GET THE DATA.

056412  012703  056614                   MOV     #TDAT00,R3              ;SEE IF THE DATA IS CORRECT.
056416  012704  056624                   MOV     #TDATI0,R4
056422  012705  000004                   MOV     #4,R5
056426  022324                   T6:     CMP     (R3)+,(R4)+
056430  001401                           BEQ     T105
056432  104000                           EMT                     ;
056434  077504                   T105:   SOB     R5,T6

                                 ;NOW TEST THE STF AC0,AC1 INSTRUCTION.

056436  012700  056574           T12:    MOV     #TPAT10,R0              ;SET UP THE INPUT DATA BUFFER.
056442  012701  056624                   MOV     #TDATI0,R1
056446  012702  000004                   MOV     #4,R2
056452  012021                   T13:    MOV     (R0)+,(R1)+
056454  077202                           SOB     R2,T13

056456  012700  056624                   MOV     #TDATI0,R0              ;SET UP AC0
056462  172410                           LDD     (R0),AC0

056464  012700  056604                   MOV     #TPAT20,R0              ;SET UP AC1
056470  172510                           LDD     (R0),AC1

056472  012701  000001                   MOV     #1,R1
056476  170001                           SETF                            ;CLEAR FD
056500  174001                   T14:    STF     AC0,AC1
056502  000240                   T15:    NOP
056504  000240                   T16:    NOP

056506  005000                           CLR     R0
056510  170200                           STFPS   R0                      ;SEE IF FPS IS CLEAR.
056512  022700  000010                   CMP     #10,R0
056516  001401                           BEQ     T17
056520  104000                           EMT                     ;
056522                           T17:
056522  170011                           SETD                            ;SET FD.

056524  012700  056614                   MOV     #TDAT00,R0
056530  174110                           STD     AC1,(R0)                ;PICK UP AC1.

056532  012737  177777  056630           MOV     #-1,@#TDATI2
056540  012737  177777  056632           MOV     #-1,@#TDATI3
056546  012703  056614                   MOV     #TDAT00,R3
```

```
056552  012704  056624           MOV     #TDATI0,R4
056556  012705  000004           MOV     #4,R5
056562  022324          T20:     CMP     (R3)+,(R4)+            ;WAS THE DATA TRANSFERRED CORRECTLY?
056564  001401                   BEQ     T23
056566  104000                   EMT                            ;
056570  077504          T23:     SOB     R5,T20
056572  000420                   BR      TDONE

056574  000000          TPAT10: 0
056576  170360          TPAT11: 170360
056600  016161          TPAT12: 016161
056602  052525          TPAT13: 052525

056604  177777          TPAT20: -1
056606  177777          TPAT21: -1
056610  177777          TPAT22: -1
056612  177777          TPAT23: -1

056614  000000          TDAT00: 0
056616  000000          TDAT01: 0
056620  000000          TDAT02: 0
056622  000000          TDAT03: 0

056624  000000          TDATI0: 0
056626  000000          TDATI1: 0
056630  000000          TDATI2: 0
056632  000000          TDATI3: 0

056634                  TDONE:
056634  004767  045732           JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                           ;SEE IF THE USER HAS EXPRESSED
                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                           ;THE USER TYPED CONTROL G?).

                        ;*****************************************************************************
                        ;TEST 442          ACCUMULATORS DATA PATTERNS TEST
                        ;*****************************************************************************
056640                  TS442:
056640  170011                   SETD                           ;SET FD.
                        ;TEST ACCUMULATOR 0 WITH FLOATING ONE
056642  012700  060374           MOV     #GPAT00,R0
056646  012701  060434           MOV     #GDAT00,R1
056652  004737  060246           JSR     PC,@#GSETUP            ;LOAD TEST PATTERN.
056656  012703  000102           MOV     #102,R3
056662                  G1:
056662  172410                   LDD     (R0),AC0
056664  174000                   STD     AC0,AC0
056666  172400                   LDD     AC0,AC0           ;STORE THE TEST PATTERN.
056670  174011                   STD     AC0,(R1)
056672  004737  060344           JSR     PC,@#GCMP              ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
056676  005737  060370           TST     @#GFLAG1
```

```
056702 001004                    BNE     G2
056704 005137 060370             COM     @#GFLAG1
056710 000261                    SEC
056712 000401                    BR      G3
056714 000241            G2:     CLC
056716 006160 000006     G3:     ROL     6(R0)                  ;GENERATE THE NEXT TEST PATTERN.
056722 006160 000004             ROL     4(R0)
056726 006160 000002             ROL     2(R0)
056732 006110                    ROL     (R0)
056734 004737 060324             JSR     PC,@#GRESET            ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
056740 077330                    SOB     R3,G1


                         ;TEST ACCUMULATOR 0 WITH FLOATING ZERO
056742 012700 060404             MOV     #GPAT10,R0
056746 012701 060434             MOV     #GDAT00,R1
056752 004737 060246             JSR     PC,@#GSETUP            ;LOAD TEST PATTERN.
056756 012703 000102             MOV     #102,R3
056762                   G4:
056762 172410                    LDD     (R0),AC0
056764 174000                    STD     AC0,AC0
056766 172400                    LDD     AC0,AC0        ;STORE THE TEST PATTERN.
056770 174011                    STD     AC0,(R1)
056772 004737 060344             JSR     PC,@#GCMP              ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
056776 005737 060370             TST     @#GFLAG1
057002 001004                    BNE     G5
057004 005137 060370             COM     @#GFLAG1
057010 000241                    CLC
057012 000401                    BR      G6
057014 000261            G5:     SEC
057016 006160 000006     G6:     ROL     6(R0)                  ;GENERATE THE NEXT TEST PATTERN.
057022 006160 000004             ROL     4(R0)
057026 006160 000002             ROL     2(R0)
057032 006110                    ROL     (R0)
057034 004737 060324             JSR     PC,@#GRESET            ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
057040 077330                    SOB     R3,G4


                         ;TEST ACCUMULATOR 1 WITH FLOATING ONE
057042 012700 060374             MOV     #GPAT00,R0
057046 012701 060434             MOV     #GDAT00,R1
057052 004737 060246             JSR     PC,@#GSETUP            ;LOAD TEST PATTERN.
057056 012703 000102             MOV     #102,R3
057062                   G7:
057062 172410                    LDD     (R0),AC0
057064 174001                    STD     AC0,AC1
057066 172401                    LDD     AC1,AC0        ;STORE THE TEST PATTERN.
057070 174011                    STD     AC0,(R1)
057072 004737 060344             JSR     PC,@#GCMP              ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
057076 005737 060370             TST     @#GFLAG1
057102 001004                    BNE     G10
057104 005137 060370             COM     @#GFLAG1
057110 000261                    SEC
057112 000401                    BR      G11
```

```
057114  000241                  G10:    CLC
057116  006160  000006          G11:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057122  006160  000004                  ROL     4(R0)
057126  006160  000002                  ROL     2(R0)
057132  006110                          ROL     (R0)
057134  004737  060324                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                        ;BUFFER.
057140  077330                          SOB     R3,G7


                                ;TEST ACCUMULATOR 1 WITH FLOATING ZERO
057142  012700  060404                  MOV     #GPAT10,R0
057146  012701  060434                  MOV     #GDAT00,R1
057152  004737  060246                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057156  012703  000102                  MOV     #102,R3
057162                          G12:
057162  172410                          LDD     (R0),AC0
057164  174001                          STD     AC0,AC1
057166  172401                          LDD     AC1,AC0         ;STORE THE TEST PATTERN.
057170  174011                          STD     AC0,(R1)
057172  004737  060344                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                        ;THAT WHICH WAS WRITTEN.
057176  005737  060370                  TST     @#GFLAG1
057202  001004                          BNE     G13
057204  005137  060370                  COM     @#GFLAG1
057210  000241                          CLC
057212  000401                          BR      G14
057214  000261                  G13:    SEC
057216  006160  000006          G14:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057222  006160  000004                  ROL     4(R0)
057226  006160  000002                  ROL     2(R0)
057232  006110                          ROL     (R0)
057234  004737  060324                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                        ;BUFFER.
057240  077330                          SOB     R3,G12


                                ;TEST ACCUMULATOR 2 WITH FLOATING ONE
057242  012700  060374                  MOV     #GPAT00,R0
057246  012701  060434                  MOV     #GDAT00,R1
057252  004737  060246                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057256  012703  000102                  MOV     #102,R3
057262                          G15:
057262  172410                          LDD     (R0),AC0
057264  174002                          STD     AC0,AC2
057266  172402                          LDD     AC2,AC0         ;STORE THE TEST PATTERN.
057270  174011                          STD     AC0,(R1)
057272  004737  060344                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                        ;THAT WHICH WAS WRITTEN.
057276  005737  060370                  TST     @#GFLAG1
057302  001004                          BNE     G16
057304  005137  060370                  COM     @#GFLAG1
057310  000261                          SEC
057312  000401                          BR      G17
057314  000241                  G16:    CLC
057316  006160  000006          G17:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057322  006160  000004                  ROL     4(R0)
057326  006160  000002                  ROL     2(R0)
```

```
057332  006110                  ROL     (R0)
057334  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
057340  077330                  SOB     R3,G15

                        ;TEST ACCUMULATOR 2 WITH FLOATING ZERO
057342  012700  060404          MOV     #GPAT10,R0
057346  012701  060434          MOV     #GDAT00,R1
057352  004737  060246          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057356  012703  000102          MOV     #102,R3
057362                  G20:
057362  172410                  LDD     (R0),AC0
057364  174002                  STD     AC0,AC2
057366  172402                  LDD     AC2,AC0         ;STORE THE TEST PATTERN.
057370  174011                  STD     AC0,(R1)
057372  004737  060344          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
057376  005737  060370          TST     @#GFLAG1
057402  001004                  BNE     G21
057404  005137  060370          COM     @#GFLAG1
057410  000241                  CLC
057412  000401                  BR      G22
057414  000261          G21:    SEC
057416  006160  000006  G22:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057422  006160  000004          ROL     4(R0)
057426  006160  000002          ROL     2(R0)
057432  006110                  ROL     (R0)
057434  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
057440  077330                  SOB     R3,G20

                        ;TEST ACCUMULATOR 3 WITH FLOATING ONE
057442  012700  060374          MOV     #GPAT00,R0
057446  012701  060434          MOV     #GDAT00,R1
057452  004737  060246          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057456  012703  000102          MOV     #102,R3
057462                  G23:
057462  172410                  LDD     (R0),AC0
057464  174003                  STD     AC0,AC3
057466  172403                  LDD     AC3,AC0         ;STORE THE TEST PATTERN.
057470  174011                  STD     AC0,(R1)
057472  004737  060344          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
057476  005737  060370          TST     @#GFLAG1
057502  001004                  BNE     G24
057504  005137  060370          COM     @#GFLAG1
057510  000261                  SEC
057512  000401                  BR      G25
057514  000241          G24:    CLC
057516  006160  000006  G25:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057522  006160  000004          ROL     4(R0)
057526  006160  000002          ROL     2(R0)
057532  006110                  ROL     (R0)
057534  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
057540  077330                  SOB     R3,G23
```

```

                                ;TEST ACCUMULATOR 3 WITH FLOATING ZERO
057542  012700  060404                  MOV     #GPAT10,R0
057546  012701  060434                  MOV     #GDAT00,R1
057552  004737  060246                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057556  012703  000102                  MOV     #102,R3
057562                          G26:
057562  172410                          LDD     (R0),AC0
057564  174003                          STD     AC0,AC3
057566  172403                          LDD     AC3,AC0         ;STORE THE TEST PATTERN.
057570  174011                          STD     AC0,(R1)
057572  004737  060344                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                        ;THAT WHICH WAS WRITTEN.
057576  005737  060370                  TST     @#GFLAG1
057602  001004                          BNE     G27
057604  005137  060370                  COM     @#GFLAG1
057610  000241                          CLC
057612  000401                          BR      G30
057614  000261                  G27:    SEC
057616  006160  000006          G30:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057622  006160  000004                  ROL     4(R0)
057626  006160  000002                  ROL     2(R0)
057632  006110                          ROL     (R0)
057634  004737  060324                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                        ;BUFFER.
057640  077330                          SOB     R3,G26


                                ;TEST ACCUMULATOR 4 WITH FLOATING ONE
057642  012700  060374                  MOV     #GPAT00,R0
057646  012701  060434                  MOV     #GDAT00,R1
057652  004737  060246                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057656  012703  000102                  MOV     #102,R3
057662                          G31:
057662  172410                          LDD     (R0),AC0
057664  174004                          STD     AC0,AC4
057666  172404                          LDD     AC4,AC0         ;STORE THE TEST PATTERN.
057670  174011                          STD     AC0,(R1)
057672  004737  060344                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                        ;THAT WHICH WAS WRITTEN.
057676  005737  060370                  TST     @#GFLAG1
057702  001004                          BNE     G32
057704  005137  060370                  COM     @#GFLAG1
057710  000261                          SEC
057712  000401                          BR      G33
057714  000241                  G32:    CLC
057716  006160  000006          G33:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
057722  006160  000004                  ROL     4(R0)
057726  006160  000002                  ROL     2(R0)
057732  006110                          ROL     (R0)
057734  004737  060324                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                        ;BUFFER.
057740  077330                          SOB     R3,G31


                                ;TEST ACCUMULATOR 4 WITH FLOATING ZERO
057742  012700  060404                  MOV     #GPAT10,R0
057746  012701  060434                  MOV     #GDAT00,R1
```

```
057752  004737  060246          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
057756  012703  000102          MOV     #102,R3
057762                  G34:
057762  172410                  LDD     (R0),AC0
057764  174004                  STD     AC0,AC4
057766  172404                  LDD     AC4,AC0         ;STORE THE TEST PATTERN.
057770  174011                  STD     AC0,(R1)
057772  004737  060344          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
057776  005737  060370          TST     @#GFLAG1
060002  001004                  BNE     G35
060004  005137  060370          COM     @#GFLAG1
060010  000241                  CLC
060012  000401                  BR      G36
060014  000261          G35:    SEC
060016  006160  000006  G36:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
060022  006160  000004          ROL     4(R0)
060026  006160  000002          ROL     2(R0)
060032  006110                  ROL     (R0)
060034  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
060040  077330                  SOB     R3,G34

                        ;TEST ACCUMULATOR 5 WITH FLOATING ONE
060042  012700  060374          MOV     #GPAT00,R0
060046  012701  060434          MOV     #GDAT00,R1
060052  004737  060246          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
060056  012703  000102          MOV     #102,R3
060062                  G37:
060062  172410                  LDD     (R0),AC0
060064  174005                  STD     AC0,AC5
060066  172405                  LDD     AC5,AC0         ;STORE THE TEST PATTERN.
060070  174011                  STD     AC0,(R1)
060072  004737  060344          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
060076  005737  060370          TST     @#GFLAG1
060102  001004                  BNE     G40
060104  005137  060370          COM     @#GFLAG1
060110  000261                  SEC
060112  000401                  BR      G41
060114  000241          G40:    CLC
060116  006160  000006  G41:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
060122  006160  000004          ROL     4(R0)
060126  006160  000002          ROL     2(R0)
060132  006110                  ROL     (R0)
060134  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
060140  077330                  SOB     R3,G37

                        ;TEST ACCUMULATOR 5 WITH FLOATING ZERO
060142  012700  060404          MOV     #GPAT10,R0
060146  012701  060434          MOV     #GDAT00,R1
060152  004737  060246          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
060156  012703  000102          MOV     #102,R3
060162                  G42:
060162  172410                  LDD     (R0),AC0
```

```
060164  174005                  STD     AC0,AC5
060166  172405                  LDD     AC5,AC0         ;STORE THE TEST PATTERN.
060170  174011                  STD     AC0,(R1)
060172  004737  060344          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                                                ;THAT WHICH WAS WRITTEN.
060176  005737  060370          TST     @#GFLAG1
060202  001004                  BNE     G43
060204  005137  060370          COM     @#GFLAG1
060210  000241                  CLC
060212  000401                  BR      G44
060214  000261          G43:    SEC
060216  006160  000006  G44:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
060222  006160  000004          ROL     4(R0)
060226  006160  000002          ROL     2(R0)
060232  006110                  ROL     (R0)
060234  004737  060324          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                ;BUFFER.
060240  077330                  SOB     R3,G42


060242  000137  060444          JMP     @#GDONE


                        ;USE THIS ROUTINE TO INITIALIZE ALL THE DATA BUFFERS.
060246  012705  060370  GSETUP: MOV     #GFLAG1,R5
060252  012704  000026          MOV     #26,R4
060256  005025          1$:     CLR     (R5)+
060260  077402                  SOB     R4,1$

060262  012705  060404          MOV     #GPAT10,R5
060266  012704  000010          MOV     #10,R4
060272  005125          2$:     COM     (R5)+
060274  077402                  SOB     R4,2$

060276  020067  000072  GS1:    CMP     R0,GPAT00
060302  001401                  BEQ     3$
060304  000207                  RTS     PC

060306  012705  060434  3$:     MOV     #GDAT00,R5
060312  012704  000004          MOV     #4,R4
060316  005125          4$:     COM     (R5)+
060320  077402                  SOB     R4,4$
060322  000207                  RTS     PC

060324  012705  060434  GRESET: MOV     #GDAT00,R5
060330  012704  000004          MOV     #4,R4
060334  005025          1$:     CLR     (R5)+
060336  077402                  SOB     R4,1$
060340  000137  060276          JMP     @#GS1


                        ;SEE IF THE DATA WRITTEN MATCHES THE DATA READ.
060344  012705  060434  GCMP:   MOV     #GDAT00,R5
060350  012704  000004          MOV     #4,R4
060354  010002                  MOV     R0,R2
060356  022225          1$:     CMP     (R2)+,(R5)+
060360  001401                  BEQ     2$
060362  104000                  EMT                     ;
```

```
060364  077404            2$:     SOB     R4,1$
060366  000207                    RTS     PC




060370  000000            GFLAG1: 0
060372  000000            GFLAG2: 0

060374  000000            GPAT00: 0
060376  000000            GPAT01: 0
060400  000000            GPAT02: 0
060402  000000            GPAT03: 0

060404  177777            GPAT10: -1
060406  177777            GPAT11: -1
060410  177777            GPAT12: -1
060412  177777            GPAT13: -1

060414  177777            GAND0:  -1
060416  177777            GAND1:  -1
060420  177777            GAND2:  -1
060422  177777            GAND3:  -1

060424  000000            GOR0:   0
060426  000000            GOR1:   0
060430  000000            GOR2:   0
060432  000000            GOR3:   0

060434  000000            GDAT00: 0
060436  000000            GDAT01: 0
060440  000000            GDAT02: 0
060442  000000            GDAT03: 0


060444                    GDONE:
060444  004767  044122            JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                          ;SEE IF THE USER HAS EXPRESSED
                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                          ;THE USER TYPED CONTROL G?).


                          ;*************************************************************************
                          ;TEST 443        FPP ACCUMULATORS DUAL ADDRESS TEST
                          ;*************************************************************************
060450                    TS443:
060450  005037  061140    H1:     CLR     @#HFLAG
060454  012700  061142            MOV     #HA1W,R0                ;INITIALIZE THE LOAD BUFFER DATA.
060460  012701  061262            MOV     #HDAT1,R1
060464  012703  000024            MOV     #24,R3
060470  012120            H2:     MOV     (R1)+,(R0)+
060472  077302                    SOB     R3,H2

060474  004767  000420            JSR     PC,HCLR         ;CLEAR THE OUTPUT DATA BUFFER.

```

```
060500  170011                  H3:     SETD
                                ;LOAD ACCUMULATOR 1
060502  012700  061142                  MOV     #HA1W,R0
060506  172410                          LDD     (R0),AC0
060510  174001                          STD     AC0,AC1

                                ;LOAD ACCUMULATOR 2
060512  012700  061152                  MOV     #HA2W,R0
060516  172410                          LDD     (R0),AC0
060520  174002                          STD     AC0,AC2

                                ;LOAD ACCUMULATOR 3
060522  012700  061162                  MOV     #HA3W,R0
060526  172410                          LDD     (R0),AC0
060530  174003                          STD     AC0,AC3

                                ;LOAD ACCUMULATOR 4
060532  012700  061172                  MOV     #HA4W,R0
060536  172410                          LDD     (R0),AC0
060540  174004                          STD     AC0,AC4

                                ;LOAD ACCUMULATOR 5
060542  012700  061202                  MOV     #HA5W,R0
060546  172410                          LDD     (R0),AC0
060550  174005                          STD     AC0,AC5

060552  004737  061006          H4:     JSR     PC,@#HSTD               ;GO READ ALL ACCUMULATORS BACK.

060556  004737  061064                  JSR     PC,@#HCMP               ;SEE IF DATA IS CORRECT.

                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 1,
                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
                                ;THE DATA.
060562  012700  061142                  MOV     #HA1W,R0
060566  012702  000004                  MOV     #4,R2
060572  010001                          MOV     R0,R1
060574  005121                  H5:     COM     (R1)+
060576  172410                          LDD     (R0),AC0
060600  174001                          STD     AC0,AC1
060602  004737  061006                  JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
060606  004737  061064                  JSR     PC,@#HCMP               ;CHECK THE DATA.
060612  077210                          SOB     R2,H5

                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 2,
                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
                                ;THE DATA.
060614  012700  061152                  MOV     #HA2W,R0
060620  012702  000004                  MOV     #4,R2
060624  010001                          MOV     R0,R1
060626  005121                  H6:     COM     (R1)+
060630  172410                          LDD     (R0),AC0
060632  174002                          STD     AC0,AC2
060634  004737  061006                  JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
060640  004737  061064                  JSR     PC,@#HCMP               ;CHECK THE DATA.
060644  077210                          SOB     R2,H6

                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 3,
                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
                                ;THE DATA.
060646  012700  061162                  MOV     #HA3W,R0
```

```
060652  012702  000004          MOV     #4,R2
060656  010001                  MOV     R0,R1
060660  005121          H7:     COM     (R1)+
060662  172410                  LDD     (R0),AC0
060664  174003                  STD     AC0,AC3
060666  004737  061006          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
060672  004737  061064          JSR     PC,@#HCMP               ;CHECK THE DATA.
060676  077210                  SOB     R2,H7

                        ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 4,
                        ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
                        ;THE DATA.
060700  012700  061172          MOV     #HA4W,R0
060704  012702  000004          MOV     #4,R2
060710  010001                  MOV     R0,R1
060712  005121          H10:    COM     (R1)+
060714  172410                  LDD     (R0),AC0
060716  174004                  STD     AC0,AC4
060720  004737  061006          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
060724  004737  061064          JSR     PC,@#HCMP               ;CHECK THE DATA.
060730  077210                  SOB     R2,H10

                        ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 5,
                        ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
                        ;THE DATA.
060732  012700  061202          MOV     #HA5W,R0
060736  012702  000004          MOV     #4,R2
060742  010001                  MOV     R0,R1
060744  005121          H11:    COM     (R1)+
060746  172410                  LDD     (R0),AC0
060750  174005                  STD     AC0,AC5
060752  004737  061006          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
060756  004737  061064          JSR     PC,@#HCMP               ;CHECK THE DATA.
060762  077210                  SOB     R2,H11


060764  005737  061140          TST     @#HFLAG
060770  001402                  BEQ     H12
060772  000137  061332          JMP     @#HDONE

060776  005137  061140  H12:    COM     @#HFLAG
061002  000137  060500          JMP     @#H3

                        ;STORE ALL ACCUMULATORS IN THE OUTPUT BUFFERS.
061006  004737  061120  HSTD:   JSR     PC,@#HCLR               ;CLEAR ALL OUTPUT BUFFERS.
                        ;STORE ACCUMULATOR 1
061012  012704  061212          MOV     #HA1R,R4
061016  172401                  LDD     AC1,AC0
061020  174014                  STD     AC0,(R4)
                        ;STORE ACCUMULATOR 2
061022  012704  061222          MOV     #HA2R,R4
061026  172402                  LDD     AC2,AC0
061030  174014                  STD     AC0,(R4)
                        ;STORE ACCUMULATOR 3
061032  012704  061232          MOV     #HA3R,R4
061036  172403                  LDD     AC3,AC0
```

```
061040  174014                          STD     AC0,(R4)
                                ;STORE ACCUMULATOR 4
061042  012704  061242                  MOV     #HA4R,R4
061046  172404                          LDD     AC4,AC0
061050  174014                          STD     AC0,(R4)
                                ;STORE ACCUMULATOR 5
061052  012704  061252                  MOV     #HA5R,R4
061056  172405                          LDD     AC5,AC0
061060  174014                          STD     AC0,(R4)
061062  000207                          RTS     PC

                                ;COMPARE DATA LOADED WITH DATA READ.
061064  012637  061136          HCMP:   MOV     (SP)+,@#HADR            ;SAVE RETURN ADDRESS.
061070  012703  061142                  MOV     #HA1W,R3
061074  012704  061212                  MOV     #HA1R,R4
061100  012705  000024                  MOV     #24,R5
061104  022324                  HCMP1:  CMP     (R3)+,(R4)+
061106  001401                          BEQ     HCMP2
061110  104000                          EMT                             ;
061112  077504                  HCMP2:  SOB     R5,HCMP1
061114  000177  000016                  JMP     @HADR

                                ;CLEAR THE DATA OUTPUT BUFFER.
061120  012704  061212          HCLR:   MOV     #HA1R,R4
061124  012705  000024                  MOV     #24,R5
061130  005024                  HCLR1:  CLR     (R4)+
061132  077502                          SOB     R5,HCLR1
061134  000207                          RTS     PC

061136  000000                  HADR:   0
061140  000000                  HFLAG:  0

061142  000000  000000  000000  HA1W:   .WORD   0,0,0,0
061150  000000
061152  000000  000000  000000  HA2W:   .WORD   0,0,0,0
061160  000000
061162  000000  000000  000000  HA3W:   .WORD   0,0,0,0
061170  000000
061172  000000  000000  000000  HA4W:   .WORD   0,0,0,0
061200  000000
061202  000000  000000  000000  HA5W:   .WORD   0,0,0,0
061210  000000

061212  000000  000000  000000  HA1R:   .WORD   0,0,0,0
061220  000000
061222  000000  000000  000000  HA2R:   .WORD   0,0,0,0
061230  000000
061232  000000  000000  000000  HA3R:   .WORD   0,0,0,0
061240  000000
061242  000000  000000  000000  HA4R:   .WORD   0,0,0,0
061250  000000
061252  000000  000000  000000  HA5R:   .WORD   0,0,0,0
061260  000000

061262  073567  073567  073567  HDAT1:  .WORD   73567,73567,73567,73567
061270  073567
```

```
061272  063146  063146  063146  HDAT2:  .WORD   63146,63146,63146,63146
061300  063146
061302  010421  010421  010421  HDAT3:  .WORD   10421,10421,10421,10421
061310  010421
061312  031463  031463  031463  HDAT4:  .WORD   31463,31463,31463,31463
061320  031463
061322  042104  042104  042104  HDAT5:  .WORD   42104,42104,42104,42104
061330  042104

061332                          HDONE:
061332  004767  043234                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************************
                                ;TEST 444       FSRC MODE 0 WITH ILLEGAL ACCUMULATOR TEST
                                ;*******************************************************************************
061336                          TS444:
061336  170011                          SETD                            ;SET FD
061340  012700  061620                  MOV     #SPAT10,R0              ;LOAD AC0
061344  172410                          LDD     (R0),AC0

061346  012737  061524  000244          MOV     #SERR0,@#FPVECT         ;USE OF THE NON-EXISTENT AC-
                                                                        ;CUMULATOR SHOULD RESULT IN
                                                                        ;A TRAP TO 244.
061354  012700  000001                  MOV     #1,R0                   ;A FAILURE IN THE FSRC FLOWS
                                                                        ;WILL RESULT IN AN ODD ADDRESS
061360  012737  061432  000004          MOV     #SERR1,@#ERRVECT        ;TRAP TO 4.
061366  005003                          CLR     R3

061370  172407                  SX2:    LDD     AC7,AC0
061372  170000                  SX3:    CFCC
061374  005203                          INC     R3
061376  005203                  SX4:    INC     R3

061400  012701  061630                  MOV     #SDAT00,R1              ;NO TRAP OCCURRED!!
061404  174011                          STD     AC0,(R1)                ;SEE IF AC0 WAS MODIFIED.

061406  012701  061630                  MOV     #SDAT00,R1
061412  012702  061620                  MOV     #SPAT10,R2
061416  012703  000004                  MOV     #4,R3
061422  022122                  SX5:    CMP     (R1)+,(R2)+
061424  001401                          BEQ     SX6
061426  104000                          EMT                     ;
061430  077304                  SX6:    SOB     R3,SX5
061432                          SERR1:
061432  104000                          EMT                     ;


                                ;NOW TEST AC6.
061434  170011                  SX7:    SETD
061436  012700  061620                  MOV     #SPAT10,R0              ;LOAD AC0
061442  172410                          LDD     (R0),AC0
061444  012737  061574  000244          MOV     #SERR4,@#FPVECT
```

```
061452  012700  000001                  MOV     #1,R0
061456  005003                          CLR     R3

061460  172406                  SX8:    LDD     AC6,AC0
061462  170000                  SX9:    CFCC
061464  005203                          INC     R3
061466  005203                  SX10:   INC     R3

061470  012701  061630                  MOV     #SDAT00,R1
061474  174011                          STD     AC0,(R1)                        ;NO TRAP! GET AC0.

061476  012701  061630                  MOV     #SDAT00,R1                      ;WAS AC0 MODIFIED.
061502  012702  061620                  MOV     #SPAT10,R2
061506  012703  000004                  MOV     #4,R3
061512  022122                  SX11:   CMP     (R1)+,(R2)+
061514  001401                          BEQ     SX12
061516  104000                          EMT                     ;
061520  077304                  SX12:   SOB     R3,SX11
061522  104000                          EMT                     ;


                                ;TRAPPED TO 244.
061524  021627  061372          SERR0:  CMP     (SP),#SX3                       ;PC OF TRAP CORRECT?
061530  001401                          BEQ     1$
061532  104000                          EMT                     ;
061534  012737  061434  061614  1$:     MOV     #SX7,@#SADR
061542  022626                  SERR10: CMP     (SP)+,(SP)+
061544  005004                          CLR     R4
061546  170204                          STFPS   R4                              ;IS FPS CORRECT?
061550  022704  100200                  CMP     #100200,R4
061554  001326                          BNE     SERR1

061556  005004                          CLR     R4
061560  170304                          STST    R4                              ;IS FEC CORRECT?
061562  022704  000002                  CMP     #2,R4
061566  001321                          BNE     SERR1
061570  000177  000020                  JMP     @SADR

061574  021627  061462          SERR4:  CMP     (SP),#SX9
061600  001401                          BEQ     1$
061602  104000                          EMT                     ;
061604  012737  061640  061614  1$:     MOV     #SDONE,@#SADR
061612  000753                          BR      SERR10

061614  000000                  SADR:   0
061616  177777                          -1
061620  010421                  SPAT10: 10421
061622  021042                  SPAT11: 21042
061624  031463                  SPAT12: 31463
061626  042104                  SPAT13: 42104

061630  000000                  SDAT00: 0
061632  000000                  SDAT01: 0
061634  000000                  SDAT02: 0
061636  000000                  SDAT03: 0

061640                          SDONE:
```

```
061640 004767 042726            JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                        ;THE USER TYPED CONTROL G?).

                         ;*******************************************************************************
                         ;TEST 445       FSRC MODE 2 TEST
                         ;*******************************************************************************
061644                   TS445:
061644                   J1:
061644 170011                   SETD                    ;SET DOUBLE MODE

061646 012700 061774            MOV     #JDAT0,R0
061652 172410                   LDD     (R0),AC0        ;LOAD AC0=ALL 1

061654 012700 061754            MOV     #JDATI0,R0
061660 005003                   CLR     R3

061662 172420            J2:    LDD     (R0)+,AC0       ;TEST INSTRUCTION
061664 005203            J3:    INC     R3
061666 005203            J4:    INC     R3

061670 012701 061764            MOV     #JDAT00,R1
061674 174011                   STD     AC0,(R1)        ;PICK UP RESULTS

061676 020027 061744            CMP     R0,#JBUF0       ;WAS AN AUTO
061702 001001                   BNE     1$
061704 104000                   EMT                     ;
061706 012702 061754     1$:    MOV     #JDATI0,R2      ;IS DATA CORRECT?
061712 012703 061764            MOV     #JDAT00,R3
061716 012704 000004            MOV     #4,R4
061722 022223            J5:    CMP     (R2)+,(R3)+
061724 001401                   BEQ     J6
061726 104000                   EMT                     ;
061730 077404            J6:    SOB     R4,J5

061732 022700 061764            CMP     #JDATI0+10,R0   ;WAS R0 INCREM.
061736 001401                   BEQ     J7
061740 104000                   EMT                     ;
061742 000420            J7:    BR      JDONE

061744 010421            JBUF0: .WORD   010421
061746 021042            JBUF1:         021042
061750 042104            JBUF2:         042104
061752 031463            JBUF3:         031463

061754 052525            JDATI0:        052525
061756 114631            JDATI1:        114631
061760 063146            JDATI2:        063146
061762 073567            JDATI3:        073567

061764 000000            JDAT00:        0
061766 000000            JDAT01:        0
061770 000000            JDAT02:        0
061772 000000            JDAT03:        0
```

```
061774  177777                  JDAT0:          -1
061776  177777                  JDAT1:          -1
062000  177777                  JDAT2:          -1
062002  177777                  JDAT3:          -1


062004                          JDONE:
062004  004767  042562                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************
                                ;TEST 446               FSRC MODE 4 TEST
                                ;*******************************************************************
062010                          TS446:
062010  170011                          SETD                    ;SET DOUBLE MODE

062012  012700  062140                  MOV     #KPAT0,R0
062016  172410                          LDD     (R0),AC0        ;LOAD A DEFAULT
                                                                ;PATTERN INTO AC0
062020  012700  062120                  MOV     #KBUF0,R0
062024  005003                          CLR     R3
062026  172440                  KX2:    LDD     -(R0),AC0       ;TEST INSTRUCTION
062030  005203                  KX3:    INC     R3
062032  005203                  KX4:    INC     R3

062034  012701  062130                  MOV     #KDAT00,R1
062040  174011                          STD     AC0,(R1)        ;PICK UP THE RESULT

062042  020027  062130                  CMP     R0,#KBUF0+10    ;WAS AN AUTO
062046  001001                          BNE     1$
062050  104000                          EMT                     ;
062052  012702  062110          1$:     MOV     #KDATI0,R2      ;IS DATA CORRECT?
062056  012703  062130                  MOV     #KDAT00,R3
062062  012704  000004                  MOV     #4,R4
062066  022223                  KX5:    CMP     (R2)+,(R3)+
062070  001401                          BEQ     KX6
062072  104000                          EMT                     ;
062074  077404                  KX6:    SOB     R4,KX5

062076  022700  062110                  CMP     #KBUF0-10,R0    ;WAS R0 DECREMENTED
062102  001401                          BEQ     KX7
062104  104000                          EMT                     ;
062106  000420                  KX7:    BR      KDONE

062110  052525                  KDATI0: .WORD   052525
062112  114631                  KDATI1:         114631
062114  063140                  KDATI2:         063140
062116  073567                  KDATI3:         073567

062120  010421                  KBUF0:          010421
062122  031463                  KBUF1:          031463
```

```
062124  042104                          KBUF2:          042104
062126  021042                          KBUF3:          021042

062130  000000                          KDAT00:         0
062132  000000                          KDAT01:         0
062134  000000                          KDAT02:         0
062136  000000                          KDAT03:         0

062140  177777                          KPAT0:          -1
062142  177777                          KPAT1:          -1
062144  177777                          KPAT2:          -1
062146  177777                          DPAT3:          -1

062150                                  KDONE:
062150  004767  042416                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).


                                        ;*******************************************************************************
                                        ;TEST 447       FSRC MODE 2, WITH FD=0, TEST
                                        ;*******************************************************************************
062154                                  TS447:
062154                                  L1:
062154  170011                                  SETD                    ;SET DOUBLE MODE

062156  012700  062302                          MOV     #LPAT10,R0
062162  172410                                  LDD     (R0),AC0        ;LOAD AC0

062164  012700  062324                          MOV     #LDATI0,R0      ;SET UP THE INPUT
062170  012701  062312                          MOV     #LPAT20,R1      ;DATA
062174  012702  000004                          MOV     #4,R2

062200  012120                          1$:     MOV     (R1)+,(R0)+
062202  077202                                  SOB     R2,1$

062204  012700  062324                          MOV     #LDATI0,R0
062210  005003                                  CLR     R3
062212  170001                                  SETF                    ;CLEAR FD.

062214  172420                          L2:     LDF     (R0)+,AC0
062216  005203                          L3:     INC     R3

062220                                  L4:
062220  170011                                  SETD                    ;SET FD

062222  012701  062336                          MOV     #LDAT00,R1
062226  174011                                  STD     AC0,(R1)        ;PICK UP RESULTS

062230  020027  062330                          CMP     R0,#LDATI2      ;WAS R0 INCREMENTED
062234  001401                                  BEQ     1$
062236  104000                                  EMT                     ;
062240  012737  177777  062330          1$:     MOV     #-1,@#LDATI2
062246  012737  177777  062332                  MOV     #-1,@#LDATI3
```

```
062254  012702  062324                  MOV     #LDATI0,R2      ;IS DATA CORRECT
062260  012703  062336                  MOV     #LDATO0,R3
062264  012704  000004                  MOV     #4,R4

062270  022223                  L5:     CMP     (R2)+,(R3)+
062272  001401                          BEQ     L6
062274  104000                          EMT                     ;
062276  077404                  L6:     SOB     R4,L5
062300  000422                          BR      LDONE

062302  177777                  LPAT10: .WORD   -1
062304  177777                  LPAT11:         -1
062306  177777                  LPAT12:         -1
062310  177777                  LPAT13:         -1

062312  052525                  LPAT20:         052525
062314  114631                  LPAT21:         114631
062316  063142                  LPAT22:         063142
062320  073567                  LPAT23:         073567
062322  000001                          .WORD   000001
062324  000000                  LDATI0:         0
062326  000000                  LDATI1:         0
062330  000000                  LDATI2:         0
062332  000000                  LDATI3:         0
062334  000001                          .WORD   00001
062336  000000                  LDATO0:         0
062340  000000                  LDATO1:         0
062342  000000                  LDATO2:         0
062344  000000                  LDATO3:         0

062346                          LDONE:
062346  004767  042220                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************************
                                ;TEST 450       FSRC MODE 2 WITH GR7, IMMEDIATE MODE, TEST
                                ;*******************************************************************************
062352                          TS450:

062352                          M1:
062352  170011                          SETD

062354  012700  062454                  MOV     #MPAT10,R0
062360  172410                          LDD     (R0),AC0        ;LOAD BACKROUND
                                                                ;PATTERN INTO AC0.
062362  005004                          CLR     R4
062364  012737  062412  000004          MOV     #MERR3,@#ERRVECT

062372  172427  000000          M15:    LDD     #0,AC0          ;TEST INSTRUCTION
        062374                          .=.-2           ;EFFECTIVELY: 05204 IS PUT IN THE FIRST
062374  005204                          .WORD   5204    ;16 BIT WORD, OR THE ''EXP-FRACTION'' WORD.
062376  005204                  M2:     INC     R4              ;NOTE THAT
```

```
062400  005204                  M3:     INC     R4              ;005204=INC R4
062402  005204                  M4:     INC     R4
062404  020427  000003                  CMP     R4,#3           ;SEE IF THE PC
062410  001401                          BEQ     M8
062412                          MERR3:
062412  104000                          EMT                     ;
062414  012700  062474          M8:     MOV     #MDAT00,R0
062420  174010                          STD     AC0,(R0)        ;GET THE DATA

062422  012700  062474                  MOV     #MDAT00,R0
062426  022720  005204                  CMP     #5204,(R0)+     ;IS THE DATA CORRECT?
062432  001401                          BEQ     M5
062434  104000                          EMT                     ;
062436  012701  000003          M5:     MOV     #3,R1
062442  005720                  M6:     TST     (R0)+
062444  001401                          BEQ     M7
062446  104000                          EMT                     ;
062450  077104                  M7:     SOB     R1,M6
062452  000414                          BR      MDONE

062454  177777                  MPAT10:         -1
062456  177777                  MPAT11:         -1
062460  177777                  MPAT12:         -1
062462  177777                  MPAT13:         -1

062464  005204                  MPAT20:         5204
062466  005204                  MPAT21:         5204
062470  005204                  MPAT22:         5204
062472  005204                  MPAT23:         5204

062474  000000                  MDAT00:         0
062476  000000                  MDAT01:         0
062500  000000                  MDAT02:         0
062502  000000                  MDAT03:         0

062504                          MDONE:
062504  004767  042062                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*********************************************************************
                                ;TEST 451       FSRC MODE 3 TEST
                                ;*********************************************************************
062510                          TS451:

062510                          N1:
062510  170011                          SETD                    ;SET FD MODE

062512  012700  062640                  MOV     #NPAT10,R0
062516  172410                          LDD     (R0),AC0        ;LOAD AC0 WITH A DEFAULT
                                                                ;PATTERN
062520  012700  062626                  MOV     #NPAT20,R0
062524  005003                          CLR     R3
```

```
062526  012737  062600  000004          MOV     #NERR0,@#ERRVECT        ;IF A FAILURE OCCURS
                                                                        ;IN THE FSRC FLOWS AN
                                                                        ;ODD TRAP TO 4 COULD OCCUR
062534  172430                  N2:     LDD     @(R0)+,AC0              ;TEST INSTRUCTION.
062536  005203                  N3:     INC     R3
062540  005203                  N4:     INC     R3

062542  012701  062606                  MOV     #NDAT00,R1
062546  174011                          STD     AC0,(R1)                ;GET THE DATA

062550  020027  062630                  CMP     R0,#NPAT20+2            ;WAS R0 INCREMENTED
062554  001401                          BEQ     N12
062556  104000                          EMT                             ;
062560  012702  062606          N12:    MOV     #NDAT00,R2              ;DATA CORRECT
062564  012703  062650                  MOV     #NDATI0,R3
062570  012704  000004                  MOV     #4,R4
062574  022223                  N13:    CMP     (R2)+,(R3)+
062576  001401                          BEQ     N14
062600                          NERR0:
062600  104000                          EMT                             ;
062602  077404                  N14:    SOB     R4,N13
062604  000425                          BR      NDONE

062606  000000                  NDAT00: .WORD   0
062610  000000                  NDAT01:         0
062612  000000                  NDAT02:         0
062614  000000                  NDAT03:         0

062616  052525  052525  052525          .WORD   52525,52525,52525,52525
062624  052525
062626  062650                  NPAT20: .WORD   NDATI0
062630  070707                  NPAT21:         070707
062632  070707                  NPAT22:         070707
062634  070707                  NPAT23:         070707
062636  000001                          .WORD   1
062640  177777                  NPAT10: .WORD   -1
062642  177777                  NPAT11:         -1
062644  177777                  NPAT12:         -1
062646  177777                  NPAT13:         -1

062650  010421                  NDATI0: .WORD   010421
062652  021042                  NDATI1:         021042
062654  031463                  NDATI2:         031463
062656  042104                  NDATI3:         042104

062660                          NDONE:
062660  004767  041706                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).



                                ;*************************************************************************
                                ;TEST 452       FSRC MODE 5 TEST
                                ;*************************************************************************
```

```
062664                          TS452:

062664                          01:
062664  170011                          SETD                    ;SET FD MODE

062666  012700  063014                  MOV     #OPAT10,R0
062672  172410                          LDD     (R0),AC0        ;LOAD AC0 WITH A
                                                                ;DEFAULT PATTERN.
062674  012700  063002                  MOV     #OPAT21,R0
062700  005003                          CLR     R3
062702  012737  062732  000004          MOV     #OERR0,@#ERRVEC ;IF A FAILURE
                                                                ;OCCURS IN THE FSRC
                                                                ;FLOWS AN ODD ADDR.
                                                                ;TRAP TO 4 MAY OCCUR.
062710  172450                  02:     LDD     @-(R0),AC0      ;TEST INSTRUCTION
062712  005203                  03:     INC     R3
062714  005203                  04:     INC     R3

062716  012701  062762                  MOV     #ODAT00,R1
062722  174011                          STD     AC0,(R1)        ;GET THE DATA

062724  020027  063000                  CMP     R0,#OPAT20      ;WAS R0 DECREMENTED
062730  001401                          BEQ     012
062732                          OERR0:
062732  104000                          EMT                     ;
062734  012702  062762          012:    MOV     #ODAT00,R2      ;DATA CORRECT?
062740  012703  063024                  MOV     #ODATI0,R3
062744  012704  000004                  MOV     #4,R4
062750  022223                  013:    CMP     (R2)+,(R3)+
062752  001401                          BEQ     014
062754  104000                          EMT                     ;
062756  077404                  014:    SOB     R4,013
062760  000425                          BR      ODONE

062762  000000                  ODAT00: .WORD   0
062764  000000                  ODAT01:         0
062766  000000                  ODAT02:         0
062770  000000                  ODAT03:         0

062772  052525  052525  052525          .WORD   52525,52525,52525
063000  063024                  OPAT20: .WORD   ODATI0
063002  070707                  OPAT21: 070707
063004  070707                  OPAT22: 070707
063006  070707                  OPAT23: 070707
063010  070707                  OPAT24: 070707
063012  000001                          .WORD   1
063014  177777                  OPAT10: .WORD   -1
063016  177777                  OPAT11:         -1
063020  177777                  OPAT12:         -1
063022  177777                  OPAT13:         -1

063024  073567                  ODATI0: .WORD   73567
063026  004210                  ODATI1:         004210
063030  114631                  ODATI2:         114631
063032  125252                  ODATI3:         125252
```

```
063034                          ODONE:
063034  004767  041532                  JSR     PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).


                                ;*****************************************************************************
                                ;TEST 453       FSRC MODE 6 TEST
                                ;*****************************************************************************
063040                          TS453:

063040                          P1:
063040  170011                          SETD                     ;SET FD MODE

063042  012700  063126                  MOV     #PPAT10,R0
063046  172410                          LDD     (R0),AC0         ;LOAD A DEFAULT PATTERN
                                                                 ;INTO AC0
063050  012700  062675                  MOV     #PDATI0-241,R0   ;COULD OCCUR.

063054  172460  000241          P2:     LDD     241(R0),AC0
        063056                  P3=P2+2

063060  012701  063146          P4:     MOV     #PDAT00,R1
063064  174011                          STD     AC0,(R1)         ;GET THE DATA
063066  012703  000004                  MOV     #4,R3
063072  012702  063136                  MOV     #PDATI0,R2
063076  012701  063146                  MOV     #PDAT00,R1
063102  022221                  P5:     CMP     (R2)+,(R1)+      ;CHECK THE DATA
063104  001401                          BEQ     2$
063106  104000                          EMT                      ;
063110  077304                  2$:     SOB     R3,P5
063112  022700  062675                  CMP     #PDATI0-241,R0   ;R0 CORRECT?
063116  001401                          BEQ     1$
063120  104000                          EMT                      ;
063122  000137  063156          1$:     JMP     @#PDONE
063126  177777                  PPAT10: .WORD   -1
063130  177777                  PPAT11:         -1
063132  177777                  PPAT12:         -1
063134  177777                  PPAT13:         -1

063136  010421                  PDATI0: .WORD   010421
063140  031463                  PDATI1:         031463
063142  052525                  PDATI2:         052525
063144  073567                  PDATI3:         073567

063146  000000                  PDAT00: .WORD   0
063150  000000                  PDAT01:         0
063152  000000                  PDAT02:         0
063154  000000                  PDAT03:         0

063156                          PDONE:
063156  004767  041410                  JSR     PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
```

```
                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                            ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************************
                                ;TEST 454         FSRC MODE 7 TEST
                                ;*******************************************************************************
063162                          TS454:

063162                          Q1:
063162  170011                          SETD

063164  012700  063250                  MOV     #QPAT10,R0
063170  172410                          LDD     (R0),AC0        ;LOAD A DEFAULT
                                                                ;PATTERN INTO AC0
063172  012700  063017                  MOV     #QPAT20-241,R0

063176  172470  000241          Q2:     LDD     @241(R0),AC0
        063200                  Q3=Q2+2

063202  012701  063270          Q4:     MOV     #QDAT00,R1
063206  174011                          STD     AC0,(R1)        ;GET THE DATA

063210  012703  000004                  MOV     #4,R3
063214  012704  063270                  MOV     #QDAT00,R4
063220  012705  063300                  MOV     #QDATI0,R5
063224  022425                  Q5:     CMP     (R4)+,(R5)+     ;CHECK THE DATA
063226  001401                          BEQ     2$
063230  104000                          EMT                     ;
063232  077304                  2$:     SOB     R3,Q5

063234  022700  063017                  CMP     #QPAT20-241,R0  ;CHECK R0.
063240  001401                          BEQ     1$
063242  104000                          EMT                     ;
063244  000137  063310          1$:     JMP     @#QDONE

063250  177777                  QPAT10: .WORD   -1
063252  177777                  QPAT11:         -1
063254  177777                  QPAT12:         -1
063256  177777                  QPAT13:         -1

063260  063300                  QPAT20: .WORD   QDATI0
063262  052525                  QPAT21:         52525
063264  052525                  QPAT22:         52525
063266  052525                  QPAT23:         52525

063270  000000                  QDAT00: .WORD   0
063272  000000                  QDAT01:         0
063274  000000                  QDAT02:         0
063276  000000                  QDAT03:         0

063300  073567                  QDATI0: .WORD   073567
063302  052525                  QDATI1: .WORD   052525
063304  031463                  QDATI2: .WORD   031463
063306  010421                  QDATI3: .WORD   010421
```

```
063310                              QDONE:
063310  004767  041256                      JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                    ;SEE IF THE USER HAS EXPRESSED
                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                    ;THE USER TYPED CONTROL G?).

                                    ;*******************************************************************************
                                    ;TEST 455       (BUT EZBT Y8),(BUT ENBT) AND (BUT FIUV) TEST
                                    ;*******************************************************************************
063314                              TS455:
063314  005037  064066                      CLR     @#UFLAG
063320  012700  064016                      MOV     #UPAT00,R0      ;SET UP AC#0 DATA.
063324  012701  000004                      MOV     #4,R1
063330  012720  177777              U0:     MOV     #-1,(R0)+
063334  077103                              SOB     R1,U0

063336  012737  000033  064070              MOV     #033,@#UTMP1
063344  012737  000023  064072              MOV     #023,@#UTMP2
063352  012737  063436  000244      U1:     MOV     #UERR0,@#FPVECT ;IN CASE (BUT FIUV FAILS)
063360  012700  000200                      MOV     #200,R0
063364  170100                              LDFPS   R0
063366  012700  064016                      MOV     #UPAT00,R0      ;LOAD AC0
063372  172410                              LDD     (R0),AC0
063374  013737  064070  064074              MOV     @#UTMP1,@#UROM1
063402  012737  000001  064076              MOV     #001,@#UROM2
063410  012737  000254  064100              MOV     #254,@#UROM3

063416  012700  064026                      MOV     #UPAT10,R0      ;LOAD 0 INTO AC0
063422  172410                      U2:     LDD     (R0),AC0
063424  012704  000204                      MOV     #204,R4         ;SEE IF FPS IS CORRECT
063430  170205                              STFPS   R5

063432  020405                              CMP     R4,R5
063434  001401                              BEQ     U3
063436                              UERR0:
063436  104000                              EMT                     ;
063440  012700  000200              U3:     MOV     #200,R0
063444  170100                              LDFPS   R0

063446  012700  064016                      MOV     #UPAT00,R0      ;LOAD AC0
063452  172410                              LDD     (R0),AC0
063454  013737  064072  064074              MOV     @#UTMP2,@#UROM1
063462  012737  000003  064076              MOV     #003,@#UROM2
063470  012737  000054  064100              MOV     #054,@#UROM3

063476  012700  064036                      MOV     #UPAT20,R0      ;LOAD A POSITIVE NUMBER
                                                                    ;INTO AC0
063502  172410                      U4:     LDD     (R0),AC0
063504  012704  000200                      MOV     #200,R4         ;FPS CORRECT?
063510  170205                              STFPS   R5

063512  020405                              CMP     R4,R5
063514  001401                              BEQ     U5
063516  104000                              EMT                     ;
063520  012700  000200              U5:     MOV     #200,R0
```

```
063524 170100                         LDFPS   R0
063526 012700 064016                  MOV     #UPAT00,R0          ;LOAD AC0
063532 172410                         LDD     (R0),AC0
063534 013737 064072 064074           MOV     @#UTMP2,@#UROM1
063542 012737 000403 064076           MOV     #403,@#UROM2
063550 012737 000056 064100           MOV     #056,@#UROM3
063556 012700 064046                  MOV     #UPAT30,R0          ;LOAD A NEGATIUE
                                                                  ;NUMBER INTO AL0
063562 172410                 U6:     LDD     (R0),AC0
063564 012704 000210                  MOV     #210,R4             ;FPS CORRECT
063570 170205                         STFPS   R5
063572 020405                         CMP     R4,R5
063574 001401                         BEQ     U7
063576 104000                         EMT                         ;
063600 012700 000200          U7:     MOV     #200,R0
063604 170100                         LDFPS   R0
063606 012700 064016                  MOV     #UPAT00,R0          ;LOAD AC0
063612 172410                         LDD     (R0),AC0
063614 013737 064070 064074           MOV     @#UTMP1,@#UROM1
063622 012737 000401 064076           MOV     #401,@#UROM2
063630 012737 000256 064100           MOV     #256,@#UROM3
063636 012700 064056                  MOV     #UPAT40,R0          ;LOAD -0 INTO AC0
063642 172410                 U10:    LDD     (R0),AC0
063644 000240                 U11:    NOP                         ;TRAP FROM HERE IF
063646 012704 000214                  MOV     #214,R4             ;SEE IF FPS IS CORRECT.
063652 170205                         STFPS   R5
063654 020405                         CMP     R4,R5
063656 001401                         BEQ     U12
063660 104000                         EMT                         ;
063662 005737 064066          U12:    TST     @#UFLAG ;SEE IF ALL THE PATTERNS
063666 001021                         BNE     U14                 ;HAVE BEEN TEST WITH
                                                                  ;BOTH AC NOT EQUAL TO 0 AND AC=0
063670 012700 064016                  MOV     #UPAT00,R0          ;IF NOT GO BACK AND
063674 012701 000004                  MOV     #4,R1               ;CHECK THEM WITH AC=0
063700 005020                 U13:    CLR     (R0)+
063702 077102                         SOB     R1,U13
063704 012737 177777 064066           MOV     #-1,@#UFLAG
063712 012737 000233 064070           MOV     #233,@#UTMP1
063720 012737 000223 064072           MOV     #223,@#UTMP2
063726 000137 063352                  JMP     @#U1
                              ;NOW SEE IF A TRAP CAN BE FORCED BY SETTING FIUV AND LOADING -0
063732 012737 063770 000244   U14:    MOV     #UERR3,@#FPVECT
063740 012700 004200                  MOV     #4200,R0            ;SET FD AND FIUV
063744 170100                         LDFPS   R0
063746 012700 064016                  MOV     #UPAT00,R0          ;SET UP AC0
063752 172410                         LDD     (R0),AC0
063754 012700 064056                  MOV     #UPAT40,R0          ;LOAD -0
063760 172410                 U15:    LDD     (R0),AC0            ;SHOULD TRAP TO 244
063762 170000                 U16:    CFCC
063764 000240                         NOP
063766 104000                         EMT                         ;

                              ;INTERRUPT HERE WHEN FIUV SET AND ATTEMPTED TO LOAD-0
063770 021627 063762          UERR3:  CMP     (SP),#U16
063774 001401                         BEQ     1$
063776 104000                         EMT                         ;
```

```
064000 022626                  1$:     CMP     (SP)+,(SP)+
064002 005000                          CLR     R0
064004 170300                          STST    R0              ;GET FEC.
064006 022700 000014                   CMP     #14,R0          ;CORRECT
064012 001433                          BEQ     UDONE
064014 104000                          EMT                     ;
064016 000000                  UPAT00: .WORD   0
064020 000000                  UPAT01:         0
064022 000000                  UPAT02:         0
064024 000000                  UPAT03:         0

064026 000000                  UPAT10: .WORD   0               ;0
064030 000000                  UPAT11:         0
064032 000000                  UPAT12:         0
064034 000000                  UPAT13:         0

064036 010421                  UPAT20: .WORD   010421          ;POS NUM
064040 114631                  UPAT21:         114631
064042 125252                  UPAT22:         125252
064044 177777                  UPAT23:         177777

064046 114631                  UPAT30:         114631          ;NEG NUM
064050 135673                  UPAT31:         135673
064052 146314                  UPAT32:         146314
064054 167356                  UPAT33:         167356

064056 100000                  UPAT40:         100000          ;NEG ZERO
064060 000000                  UPAT41:         0
064062 000000                  UPAT42:         0
064064 000000                  UPAT43:         0

064066 000000                  UFLAG:  .WORD   0
064070 000000                  UTMP1:          0
064072 000000                  UTMP2:          0
064074 000000                  UROM1:          0
064076 000000                  UROM2:          0
064100 000000                  UROM3:          0
064102                         UDONE:



                               ;*****************************************************************************
                               ;TEST 456        ADDF,ADDD,SUBF AND SUBD WITH FSRC=AC=0 TEST
                               ;*****************************************************************************
064102                         TS456:
064102 012700 000200                   MOV     #200,R0
064106 170100                          LDFPS   R0              ;SET DOUBLE MODE
064110 012700 064432                   MOV     #WPAT00,R0      ;LOAD AC0=:
064114 172410                          LDD     (R0),AC0
064116 012700 064432                   MOV     #WPAT00,R0
064122 172010                  W2:     ADDD    (R0),AC0        ;TEST INSTRUCTION. ADD ITSELF
064124 170205                          STFPS   R5              ;GET FPS
```

```
064126  170011                   SETD                     ;SET DOUBLE MODE
064130  012700  064432           MOV    #WPAT00,R0
064134  174010                   STD    AC0,(R0)          ;GET THE RESULT
064136  012701  064432           MOV    #WPAT00,R1
064142  012702  000004           MOV    #4,R2
064146  022021          W3:      CMP    (R0)+,(R1)+       ;IS RESULT CORRECT
064150  001401                   BEQ    W4
064152  104000                   EMT                      ;
064154  077204          W4:      SOB    R2,W3
064156  022705  000204           CMP    #204,R5           ;IS FPS CORRECT
064162  001401                   BEQ    W5
064164  104000                   EMT                      ;
064166  012700  000200  W5:      MOV    #200,R0
064172  170100                   LDFPS  R0                ;SET DOUBLE MODE
064174  012700  064432           MOV    #WPAT00,R0        ;LOAD AC0=0
064200  172410                   LDD    (R0),AC0
064202  005000                   CLR    R0
064204  170100                   LDFPS  R0                ;GO TO FLOATING MODE
064206  012700  064432           MOV    #WPAT00,R0
064212  172010          W6:      ADDF   (R0),AC0          ;TEST INSTRUCTION
064214  170205                   STFPS  R5                ;GET FPS
064216  170011                   SETD                     ;RESET TO DOUBLE MODE
064220  012700  064432           MOV    #WPAT00,R0
064224  174010                   STD    AC0,(R0)          ;GET THE RESULT
064226  012701  064432           MOV    #WPAT00,R1
064232  012702  000004           MOV    #4,R2
064236  022021          W7:      CMP    (R0)+,(R1)+       ;WAS THE RESULT
064240  001401                   BEQ    W10
064242  104000                   EMT                      ;
064244  077204          W10:     SOB    R2,W7
064246  022705  000004           CMP    #4,R5             ;WAS FPS CORRECT
064252  001401                   BEQ    W11
064254  104000                   EMT                      ;
064256  012700  000200  W11:     MOV    #200,R0
064262  170100                   LDFPS  R0                ;SET DOUBLE MODE
064264  012700  064432           MOV    #WPAT00,R0        ;LOAD AC0=0
064270  172410                   LDD    (R0),AC0
064272  012700  064432           MOV    #WPAT00,R0
064276  173010          W12:     SUBD   (R0),AC0          ;TEST INSTRUCTION
064300  170205                   STFPS  R5                ;GET FPS
064302  170011                   SETD                     ;SET DOUBLE MODE
064304  012700  064432           MOV    #WPAT00,R0
064310  174010                   STD    AC0,(R0)          ;GET THE RESULT
064312  012701  064432           MOV    #WPAT00,R1
064316  012702  000004           MOV    #4,R2
064322  022021          W13:     CMP    (R0)+,(R1)+       ;IS RESULT CORRECT?
064324  001401                   BEQ    W14
064326  104000                   EMT                      ;
064330  077204          W14:     SOB    R2,W13
064332  022705  000204           CMP    #204,R5           ;IS FPS CORRECT?
064336  001401                   BEQ    W15
064340  104000                   EMT                      ;
064342  012700  000200  W15:     MOV    #200,R0
064346  170100                   LDFPS  R0                ;SET DOUBLE MODE
064350  012700  064432           MOV    #WPAT00,R0        ;LOAD AC0=0
064354  172410                   LDD    (R0),AC0
```

```
064356  005000                          CLR     R0
064360  170100                          LDFPS   R0                 ;ENTER FLOATING MODE.
064362  012700  064432                  MOV     #WPAT00,R0
064366  173010                  W16:    SUBF    (R0),AC0           ;TEST INSTRUCTION.
064370  170205                          STFPS   R5                 ;GET FPS
064372  170011                          SETD                       ;RESET TO DOUBLE MODE
064374  012700  064432                  MOV     #WPAT00,R0         ;GET THE RESULT.
064400  174010                          STD     AC0,(R0)
064402  012701  064432                  MOV     #WPAT00,R1
064406  012702  000004                  MOV     #4,R2
064412  022021                  W17:    CMP     (R0)+,(R1)+        ;IS RESULT CORRECT?
064414  001401                          BEQ     W20
064416  104000                          EMT                        ;
064420  077204                  W20:    SOB     R2,W17
064422  022705  000004                  CMP     #4,R5              ;IS FPS CORRECT?
064426  001411                          BEQ     WDONE
064430  104000                          EMT                        ;

064432  000000                  WPAT00: .WORD   0
064434  000000                  WPAT01:         0
064436  000000                  WPAT02:         0
064440  000000                  WPAT03:         0

064442  000000                  WDAP00: .WORD   0
064444  000000                  WDAT01:         0
064446  000000                  WDAT02:         0
064450  000000                  WDAT03:         0

064452                          WDONE:
064452  004767  040114                  JSR     PC,.RSET           ;GO INITIALIZE THE FPS AND STACK; AND
                                                                   ;SEE IF THE USER HAS EXPRESSED
                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                   ;THE USER TYPED CONTROL G?).



                                ;*******************************************************************************
                                ;TEST 457       ADDD AND SUB WITH FSRC=0
                                ;*******************************************************************************
064456                          TS457:
064456  012700  000200                  MOV     #200,R0
064462  170100                          LDFPS   R0                 ;SET DOUBLE MODE
064464  012700  065016                  MOV     #XPAT00,R0         ;SET AC0 TO POSITIVE
064470  172410                          LDD     (R0),AC0
064472  012700  065026                  MOV     #XPAT10,R0         ;FSRC=0
064476  172010                  X2:     ADDD    (R0),AC0           ;TEST INSTRUCTION
064500  170205                          STFPS   R5
064502  170011                          SETD
064504  012700  065006                  MOV     #XDAT00,R0         ;GET RESULT.
064510  174010                          STD     AC0,(R0)
064512  012701  065016                  MOV     #XPAT00,R1
064516  012702  000004                  MOV     #4,R2
064522  022021                  X3:     CMP     (R0)+,(R1)+        ;IS RESULT CORRECT?
064524  001401                          BEQ     X4
064526  104000                          EMT                        ;
064530  077204                  X4:     SOB     R2,X3
```

```
064532  012704  000200          MOV     #200,R4
064536  020405                  CMP     R4,R5           ;IS FPS CORRECT?
064540  001401                  BEQ     X5
064542  104000                  EMT                     ;
064544  012700  000200  X5:     MOV     #200,R0
064550  170100                  LDFPS   R0              ;SET DOUBLE MODE
064552  012700  065036          MOV     #XPAT20,R0      ;SET AC0 TO
064556  172410                  LDD     (R0),AC0
064560  012700  065026          MOV     #XPAT10,R0      ;FSRC=0
064564  172010          X6:     ADDD    (R0),AC0        ;TEST INSTRUCTION
064566  170205                  STFPS   R5
064570  170011                  SETD
064572  012700  065006          MOV     #XDAT00,R0      ;GET RESULT
064576  174010                  STD     AC0,(R0)
064600  012701  065036          MOV     #XPAT20,R1
064604  012702  000004          MOV     #4,R2
064610  022021          X7:     CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
064612  001401                  BEQ     X10
064614  104000                  EMT                     ;
064616  077204          X10:    SOB     R2,X7
064620  012704  000210          MOV     #210,R4
064624  020405                  CMP     R4,R5           ;IS FPS CORRECT?
064626  001401                  BEQ     X11
064630  104000                  EMT                     ;
064632  012700  000200  X11:    MOV     #200,R0
064636  170100                  LDFPS   R0              ;SET DOUBLE MODE
064640  012700  065016          MOV     #XPAT00,R0      ;SET AC0 TO NON-ZERO
064644  172410                  LDD     (R0),AC0
064646  012700  065026          MOV     #XPAT10,R0      ;FSRC=0
064652  173010          X12:    SUBD    (R0),AC0        ;TEST INSTRUCTION
064654  170205                  STFPS   R5
064656  170011                  SETD
064660  012700  065006          MOV     #XDAT00,R0      ;GET RESULT
064664  174010                  STD     AC0,(R0)
064666  012701  065016          MOV     #XPAT00,R1
064672  012702  000004          MOV     #4,R2
064676  022021          X13:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
064700  001401                  BEQ     X14
064702  104000                  EMT                     ;
064704  077204          X14:    SOB     R2,X13
064706  012704  000200          MOV     #200,R4         ;IS FPS CORRECT?
064712  020405                  CMP     R4,R5
064714  001401                  BEQ     X15
064716  104000                  EMT                     ;
064720  012700  000200  X15:    MOV     #200,R0
064724  170100                  LDFPS   R0              ;SET DOUBLE MODE
064726  012700  065036          MOV     #XPAT20,R0      ;SET AC0=A NEGATIVE
064732  172410                  LDD     (R0),AC0
064734  012700  065026          MOV     #XPAT10,R0      ;FSRC=0
064740  173010          X16:    SUBD    (R0),AC0        ;TEST INSTRUCTION.
064742  170205                  STFPS   R5
064744  170011                  SETD
064746  012700  065006          MOV     #XDAT00,R0      ;GET RESULT
064752  174010                  STD     AC0,(R0)
064754  012701  065036          MOV     #XPAT20,R1
064760  012702  000004          MOV     #4,R2
```

```
064764  022021                   X17:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
064766  001401                           BEQ     X20
064770  104000                           EMT                     ;
064772  077204                   X20:    SOB     R2,X17
064774  012704  000210                   MOV     #210,R4         ;IS FPS CORRECT?
065000  020405                           CMP     R4,R5
065002  001421                           BEQ     XDONE
065004  104000                           EMT                     ;
065006  000000                   XDAT00: .WORD   0
065010  000000                   XDAT01:         0
065012  000000                   XDAT02:         0
065014  000000                   XDAT03:         0

065016  010421                   XPAT00: .WORD   010421
065020  021042                   XPAT01:         021042
065022  031463                   XPAT02:         031463
065024  042104                   XPAT03:         042104

065026  000000                   XPAT10: .WORD   0
065030  000000                   XAPT11:         0
065032  000000                   XPAT12:         0
065034  000000                   XPAT13:         0
065036  104210                   XPAT20: .WORD   104210
065040  114631                   XPAT21:         114631
065042  125252                   XPAT22:         125252
065044  135673                   XPAT23:         135673

065046                           XDONE:
065046  004767  037520                   JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).

                                 ;*********************************************************************
                                 ;TEST 460        SUBD WITH AC=0 TEST
                                 ;*********************************************************************
065052                           TS460:
065052  005037  065226                   CLR     @#YFLAG
065056  012737  065246  065230           MOV     #YPAT00,@#YTMP1 ;P
065064  012737  065256  065232           MOV     #YPAT10,@#YTMP2 ;N
065072  012737  000210  065234           MOV     #210,@#YTMP3
065100  012700  000200           Y1:     MOV     #200,R0
065104  170100                           LDFPS   R0              ;SET DOUBLE MODE
065106  012700  065266                   MOV     #YPAT20,R0      ;SET AC0=0
065112  172410                           LDD     (R0),AC0
065114  013700  065230                   MOV     @#YTMP1,R0
065120  173010                   Y2:     SUBD    (R0),AC0        ;TEST INSTRUCTION
065122  170205                           STFPS   R5
065124  170011                           SETD
065126  012700  065236                   MOV     #YDAT00,R0      ;GET RESULT
065132  174010                           STD     AC0,(R0)
065134  012702  000004                   MOV     #4,R2
065140  013701  065232                   MOV     @#YTMP2,R1      ;CHECK RESULT.
065144  022021                   Y3:     CMP     (R0)+,(R1)+
065146  001401                           BEQ     1$
```

```
065150  104000                          EMT                             ;
065152  077204                  1$:     SOB     R2,Y3
065154  023705  065234                  CMP     @#YTMP3,R5              ;FPS CORRECT?
065160  001401                          BEQ     Y4
065162  104000                          EMT                             ;
065164  005737  065226          Y4:     TST     @#YFLAG                 ;FINISHED TEST?
065170  001015                          BNE     Y5
065172  012737  177777  065226          MOV     #-1,@#YFLAG
065200  012737  065256  065230          MOV     #YPAT10,@#YTMP1
065206  012737  065246  065232          MOV     #YPAT00,@#YTMP2
065214  012737  000200  065234          MOV     #200,@#YTMP3
065222  000726                          BR      Y1
065224  000424                  Y5:     BR      YDONE

065226  000000                  YFLAG:  .WORD   0
065230  000000                  YTMP1:          0
065232  000000                  YTMP2:          0
065234  000000                  YTMP3:          0

065236  000000                  YDAT00: .WORD   0
065240  000000                  YDAT01:         0
065242  000000                  YDAT02:         0
065244  000000                  YDAT03:         0

065246  063146                  YPAT00:         063146
065250  052525                  YPAT01:         052525
065252  042104                  YPAT02:         042104
065254  167356                  YPAT03:         167356

065256  163146                  YPAT10:         163146
065260  052525                  YPAT11:         052525
065262  042104                  YPAT12:         042104
065264  167356                  YPAT13:         167356

065266  000000                  YPAT20:         0
065270  000000                  YPAT21:         0
065272  000000                  YPAT22:         0
065274  000000                  YPAT23:         0

065276                          YDONE:
065276  004767  037270                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).
                                ;*****************************************************************************
                                ;TEST 461       ADDD WITH AC=0 TEST
                                ;*****************************************************************************
065302                          TS461:
065302  005067  000134                  CLR     ZFLAG
065306  012737  065460  065444          MOV     #ZPAT00,@#ZTMP1 ;P
065314  012737  000200  065446          MOV     #200,@#ZTMP2
065322  012700  000200          Z1:     MOV     #200,R0
065326  170100                          LDFPS   R0                      ;SET DOUBLE MODE
065330  012700  065500                  MOV     #ZPAT20,R0              ;SET AC0=0
065334  172410                          LDD     (R0),AC0
```

```
065336  013700  065444                      MOV     @#ZTMP1,R0
065342  172010                      Z2:     ADDD    (R0),AC0        ;TEST INSTRUCTION
065344  170205                              STFPS   R5
065346  170011                              SETD
065350  012700  065450                      MOV     #ZDAT00,R0      ;GET RESULT
065354  174010                              STD     AC0,(R0)
065356  012702  000004                      MOV     #4,R2
065362  013701  065444                      MOV     @#ZTMP1,R1      ;RESULT CORRECT?
065366  022021                      Z3:     CMP     (R0)+,(R1)+
065370  001401                              BEQ     Z4
065372  104000                              EMT                     ;
065374  077204                      Z4:     SOB     R2,Z3
065376  023705  065446                      CMP     @#ZTMP2,R5      ;FPS CORRECT?
065402  001401                              BEQ     Z5
065404  104000                              EMT                     ;
065406  005737  065442              Z5:     TST     @#ZFLAG         ;FINISHED TEST?
065412  001012                              BNE     Z6
065414  012737  177777  065442              MOV     #-1,@#ZFLAG
065422  012737  065470  065444              MOV     #ZPAT10,@#ZTMP1
065430  012737  000210  065446              MOV     #210,@#ZTMP2
065436  000731                              BR      Z1
065440  000423                      Z6:     BR      ZDONE

065442  000000                      ZFLAG:  .WORD   0
065444  000000                      ZTMP1:          0
065446  000000                      ZTMP2:          0

065450  000000                      ZDAT00: .WORD   0
065452  000000                      ZDAT01:         0
065454  000000                      ZDAT02:         0
065456  000000                      ZDAT03:         0

065460  031463                      ZPAT00:         031463
065462  010421                      ZPAT01:         010421
065464  146314                      ZPAT02:         146314
065466  156735                      ZPAT03:         156735

065470  156735                      ZPAT10:         156735
065472  167356                      ZPAT11:         167356
065474  135673                      ZPAT12:         135673
065476  146314                      ZPAT13:         146314

065500  000000                      ZPAT20:         0
065502  000000                      ZPAT21:         0
065504  000000                      ZPAT22:         0
065506  000000                      ZPAT23:         0

065510                              ZDONE:
065510  004767  037056                      JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                    ;SEE IF THE USER HAS EXPRESSED
                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                    ;THE USER TYPED CONTROL G?).


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

```
                                 ;TEST 462       ADDF AND ADDD WITH E(AC)=E(FSRC) TEST AND (BUT FT) TEST
                                 ;*******************************************************************************
065514                           TS462:
065514  012700  003240                   MOV     #3240,R0
065520  170100                           LDFPS   R0              ;SET FIU FIV FD AND FT
065522  012700  065724                   MOV     #AAPAT0,R0      ;FLOWS IN TRAP WILL
                                                                 ;OCCUR
065526  172410                           LDD     (R0),AC0        ;SET UP AC0
065530  012700  065734                   MOV     #AAPAT1,R0
065534  172010                   AA2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
                                                                 ;SHOULD TRUNCATE
065536  012700  065714           AA3:    MOV     #AADAT0,R0
065542  174010                           STD     AC0,(R0)        ;GET THE RESULT
065544  012701  065744                   MOV     #AAPAT2,R1
065550  012702  000004                   MOV     #4,R2
065554  022021                   AA4:    CMP     (R0)+,(R1)+     ;CORRECT?
065556  001401                           BEQ     AA7
065560  104000                           EMT                     ;
065562  077204                   AA7:    SOB     R2,AA4

                                 ;NOW TEST DOUBLE FLOATING ROUND MODE.
                                 ;A 1 SHOULD BE ADDED TO THE LSB ON ROUNG MODE.

065564  012700  003200                   MOV     #3200,R0        ;SET FD FIV FIV. FT=0
065570  170100                           LDFPS   R0
065572  012700  065724                   MOV     #AAPAT0,R0
065576  172410                           LDD     (R0),AC0        ;SET UP AC0 OPERAND
065600  012700  065734                   MOV     #AAPAT1,R0
065604  172010                   AA11:   ADDD    (R0),AC0        ;TEST INSTRUCTION
                                                                 ;SHOULD ROUND
065606  012700  065714           AA12:   MOV     #AADAT0,R0
065612  174010                           STD     AC0,(R0)        ;GET THE RESULT
065614  012701  065754                   MOV     #AAPAT3,R1
065620  012702  000004                   MOV     #4,R2
065624  022021                   AA13:   CMP     (R0)+,(R1)+     ;CORRECT?
065626  001401                           BEQ     AA20
065630  104000                           EMT                     ;
065632  077204                   AA20:   SOB     R2,AA13

                                 ;NOW TEST ADDF WITH FT=0, ROUND MODE

065634  012700  003200                   MOV     #3200,R0                ;FIV=1, FIV=1, FT=0
065640  170100                           LDFPS   R0
065642  012700  065724                   MOV     #AAPAT0,R0      ;LOAD AC0 OPERAND
065646  172410                           LDD     (R0),AC0
065650  170001                           SETF                    ;ENTER FLOATING MODE
065652  012700  065774                   MOV     #AAPAT5,R0
065656  172010                   AA22:   ADDF    (R0),AC0        ;TEST INSTRUCTION
                                                                 ;SHOULD ROUND
065660                           AA23:
065660  170011                           SETD                    ;RESET TO DOUBLE
                                                                 ;MODE
065662  012700  065714                   MOV     #AADAT0,R0      ;GET THE RESULT
065666  174010                           STD     AC0,(R0)
065670  012701  066004                   MOV     #AAPAT6,R1      ;CORRECT?
065674  012702  000002                   MOV     #2,R2
```

```
065700  022021            AA24:   CMP     (R0)+,(R1)+
065702  001401                    BEQ     AA27
065704  104000                    EMT                             ;
065706  077204            AA27:   SOB     R2,AA24
065710  000137  066014            JMP     @#AADONE
065714  000000            AADAT0:         0
065716  000000                            0
065720  000000                            0
065722  000000                            0
065724  000200            AAPAT0:         200
065726  000000                            0
065730  000000                            0
065732  000000                            0
065734  000200            AAPAT1:         200
065736  000000                            0
065740  000000                            0
065742  000001                            1
065744  000400            AAPAT2:         400
065746  000000                            0
065750  000000                            0
065752  000000                            0
065754  000400            AAPAT3:         400
065756  000000                            0
065760  000000                            0
065762  000001                            1
065764  000400            AAPAT4:         400
065766  000000                            0
065770  100000                            100000
065772  000000                            0
065774  000200            AAPAT5:         200
065776  000001                            1
066000  000000                            0
066002  000000                            0
066004  000400            AAPAT6:         400
066006  000001                            1
066010  000000                            0
066012  000000                            0
066014                    AADONE:
066014  004767  036552            JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                          ;SEE IF THE USER HAS EXPRESSED
                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                          ;THE USER TYPED CONTROL G?).
                          ;*************************************************************************
                          ;TEST 463        ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC) TEST
                          ;*************************************************************************
066020                    TS463:
                          ;EXPONENT DIFFERENCE=57=71 (OCT) FD=1
066020  012704  003200            MOV     #3200,R4        ;SET FIV,FIV, AND FD
066024  170104                    LDFPS   R4
066026  012700  066474            MOV     #CCP0,R0        ;SET AC0 OPERAND
066032  172410                    LDD     (R0),AC0                ;AC0
066034  012700  066514            MOV     #CCP2,R0
066040  172010            CCX2:   ADDD    (R0),AC0        ;TEST INSTRUCTION
066042  170205                    STFPS   R5              ;GET FPS
066044  012700  066464            MOV     #CCDAT0,R0      ;GET THE RESULT
```

```
066050 174010                    STD     AC0,(R0)
066052 012701 066514             MOV     #CCP2,R1         ;IS IT CORRECT
066056 012702 000004             MOV     #4,R2
066062 022021            CCX3:   CMP     (R0)+,(R1)+
066064 001401                    BEQ     CCX6
066066 104000                    EMT                      ;
066070 077204            CCX6:   SOB     R2,CCX3
066072 020405                    CMP     R4,R5            ;FPS CORRECT?
066074 001401                    BEQ     CCX7
066076 104000                    EMT                      ;
                         ;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
066100 012704 003200     CCX7:   MOV     #3200,R4         ;SET FIV,FIV, AND FD
066104 170104                    LDFPS   R4
066106 012700 066474             MOV     #CCP0,R0         ;SET AC0 OPERAND
066112 172410                    LDD     (R0),AC0
066114 012700 066504             MOV     #CCP1,R0         ;FSRC
066120 172010            CCX8:   ADDD    (R0),AC0         ;TEST INSTRUCTION
066122 170205                    STFPS   R5               ;GET FPS
066124 012700 066464             MOV     #CCDAT0,R0       ;GET THE RESULT
066130 174010                    STD     AC0,(R0)
066132 012701 066564             MOV     #CCP7,R1         ;IS IT CORRECT
066136 012702 000004             MOV     #4,R2
066142 022021            CCX9:   CMP     (R0)+,(R1)+
066144 001401                    BEQ     CCX12
066146 104000                    EMT                      ;
066150 077204            CCX12:  SOB     R2,CCX9
066152 020405                    CMP     R4,R5            ;FPS CORRECT?
066154 001401                    BEQ     CCX13
066156 104000                    EMT                      ;
                         ;EXPONENT DIFFERENCE=25=31 (OCT) FD=0
066160 012700 066474     CCX13:  MOV     #CCP0,R0         ;SET UP AC0 OPERAND.
066164 172410                    LDD     (R0),AC0
066166 012704 003000             MOV     #3000,R4         ;SET FIV,FIV. CLEAR FD.
066172 170104                    LDFPS   R4
066174 012700 066554             MOV     #CCP6,R0         ;FSRC
066200 172010            CCX14:  ADDF    (R0),AC0         ;TEST INSTRUCTION
066202 170205                    STFPS   R5
066204 170011                    SETD                     ;REENTER DOUBLE MOVE
066206 012700 066464             MOV     #CCDAT0,R0       ;GET THE RESULT
066212 174010                    STD     AC0,(R0)
066214 012701 066554             MOV     #CCP6,R1         ;IS THE RESULT CORRECT?
066220 012702 000002             MOV     #2,R2
066224 022021            CCX15:  CMP     (R0)+,(R1)+
066226 001401                    BEQ     CCX18
066230 104000                    EMT                      ;
066232 077204            CCX18:  SOB     R2,CCX15
066234 020405                    CMP     R4,R5
066236 001401                    BEQ     CCX19
066240 104000                    EMT                      ;
                         ;EXPONENT DIFFERENCE=24=30 (OCT) FD=0
066242 012700 066524     CCX19:  MOV     #CCP3,R0         ;SET UP AC0 OPERAND.
066246 172410                    LDD     (R0),AC0
066250 012704 003000             MOV     #3000,R4         ;SET FIV,FIV. CLEAR FD.
066254 170104                    LDFPS   R4
066256 012700 066544             MOV     #CCP5,R0         ;FSRC
066262 172010            CCX20:  ADDF    (R0),AC0         ;TEST INSTRUCTION
```

```
066264 170205                          STFPS   R5
066266 170011                          SETD                    ;REENTER DOUBLE MOVE
066270 012700 066464                   MOV     #CCDATO,R0      ;GET THE RESULT
066274 174010                          STD     AC0,(R0)
066276 012701 066574                   MOV     #CCP10,R1       ;IS THE RESLT CORRECT?
066302 012702 000002                   MOV     #2,R2
066306 022021                 CCX21:   CMP     (R0)+,(R1)+
066310 001401                          BEQ     CCX24
066312 104000                          EMT                     ;
066314 077204                 CCX24:   SOB     R2,CCX21
066316 020405                          CMP     R4,R5
066320 001401                          BEQ     CCX25
066322 104000                          EMT                     ;
                              ;EXPONENT DIFFERENCE=1 FD=1
066324 012704 003200          CCX25:   MOV     #3200,R4        ;SET FIV,FIV, AND FD
066330 170104                          LDFPS   R4
066332 012700 066474                   MOV     #CCP0,R0        ;SET AC0 OPERAND
066336 172410                          LDD     (R0),AC0
066340 012700 066524                   MOV     #CCP3,R0        ;FSRC
066344 172010                 CCX26:   ADDD    (R0),AC0        ;TEST INSTRUCTION
066346 170205                          STFPS   R5              ;GET FPS
066350 012700 066464                   MOV     #CCDATO,R0      ;GET THE RESULT
066354 174010                          STD     AC0,(R0)
066356 012701 066604                   MOV     #CCP11,R1       ;IS IT CORRECT
066362 012702 000004                   MOV     #4,R2
066366 022021                 CCX27:   CMP     (R0)+,(R1)+
066370 001401                          BEQ     CCX30
066372 104000                          EMT                     ;
066374 077204                 CCX30:   SOB     R2,CCX27
066376 020405                          CMP     R4,R5           ;FPS CORRECT?
066400 001401                          BEQ     CCX31
066402 104000                          EMT                     ;
                              ;EXPONENT DIFFERENCE=100=144 (OCT) FD=1
066404 012704 003200          CCX31:   MOV     #3200,R4        ;SET FIV,FIV, AND FD
066410 170104                          LDFPS   R4
066412 012700 066474                   MOV     #CCP0,R0        ;SET AC0 OPERAND
066416 172410                          LDD     (R0),AC0
066420 012700 066534                   MOV     #CCP4,R0        ;FSRC
066424 172010                 CCX32:   ADDD    (R0),AC0        ;TEST INSTRUCTION
066426 170205                          STFPS   R5              ;GET FPS
066430 012700 066464                   MOV     #CCDATO,R0      ;GET THE RESULT
066434 174010                          STD     AC0,(R0)
066436 012701 066534                   MOV     #CCP4,R1        ;IS IT CORRECT
066442 012702 000004                   MOV     #4,R2
066446 022021                 CCX33:   CMP     (R0)+,(R1)+
066450 001401                          BEQ     CCX36
066452 104000                          EMT                     ;
066454 077204                 CCX36:   SOB     R2,CCX33
066456 020405                          CMP     R4,R5           ;FPS CORRECT?
066460 001461                          BEQ     CCXDONE
066462 104000                          EMT                     ;
066464 000000                 CCDATO:  0
066466 000000                          0
066470 000000                          0
066472 000000                          0
066474 000200                 CCP0:    200                     ;E(AC)=1
```

```
066476  000000                          0
066500  000000                          0
066502  000000                          0
066504  016200                  CCP1:   16200                   ;E(FSRC)=E(AC)+56=57
066506  000000                          0                       ;                  =71(OCT)
066510  000000                          0
066512  000000                          0
066514  016400                  CCP2:   16400                   ;E(FSRC)=E(AC)+57=58
066516  000000                          0                       ;                  =72(OCT)
066520  000000                          0
066522  000000                          0
066524  000400                  CCP3:   400                     ;E(FSRC)=E(AC)+1=2
066526  000000                          0
066530  000000                          0
066532  000000                          0
066534  031200                  CCP4:   31200                   ;E(FSRC)=E(AC)+100=101=145(OCT)
066536  000000                          0
066540  000000                          0
066542  000000                          0
066544  006200                  CCP5:   6200                    ;E(FSRC)=E(AC)+24=25=31(OCT)
066546  000000                          0
066550  000000                          0
066552  000000                          0
066554  006400                  CCP6:   6400                    ;E(FSRC)=E(AC)+25=26=32(OCT)
066556  000000                          0
066560  000000                          0
066562  000000                          0
066564  016200                  CCP7:   16200                   ;CCP1 RES
066566  000000                          0
066570  000000                          0
066572  000001                          1
066574  006200                  CCP10:  6200                    ;CCP5 RES
066576  000001                          1
066600  000000                          0
066602  000000                          0
066604  000500                  CCP11:  500                     ;CCP3 RES
066606  000000                          0
066610  000000                          0
066612  000000                          0
066614  000200                  CCP12:  200                     ;BAD CONSTANT
066616  000000                          0                       ;RES CCP2,CCP4
066620  000000                          0
066622  000000                          0

066624                          CCXDONE:
066624  004767  035742                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 464       ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC) TEST
                                ;*******************************************************************************
066630                          TS464:
                                ;EXPONENT DIFFERENCE=57=71 (OCT) FD=1
```

```
066630  012704  003200              MOV     #3200,R4        ;SET FIV FIV, AND FD
066634  170104                      LDFPS   R4
066636  012700  067324              MOV     #BBPAT2,R0      ;SET AC0 OPERAND.
066642  172410                      LDD     (R0),AC0
066644  012700  067314              MOV     #BBPAT1,R0      ;FSRC
066650  172010              BB2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
066652  170205                      STFPS   R5
066654  012700  067274      BB3:    MOV     #BBDAT0,R0      ;GET THE RESULT
066660  174010                      STD     AC0,(R0)
066662  012701  067324              MOV     #BBPAT2,R1      ;RESULT CORRECT?
066666  012702  000004              MOV     #4,R2
066672  022021              BB4:    CMP     (R0)+,(R1)+
066674  001401                      BEQ     BB5
066676  104000                      EMT                     ;
066700  077204              BB5:    SOB     R2,BB4
                                                            ;WAS FPS CORRECT?
066702  020405                      CMP     R4,R5
066704  001401                      BEQ     BB6
066706  104000                      EMT                     ;
                            ;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
066710  012704  003200      BB6:    MOV     #3200,R4        ;SET FIV,FIV, AND FD
066714  170104                      LDFPS   R4
066716  012700  067344              MOV     #BBPAT4,R0      ;SET AC0 OPERAND
066722  172410                      LDD     (R0),AC0
066724  012700  067314              MOV     #BBPAT1,R0      ;FSRC
066730  172010              BB7:    ADDD    (R0),AC0        ;TEST INSTRUCTION
066732  170205                      STFPS   R5              ;GET FPS
066734  012700  067274              MOV     #BBDAT0,R0      ;GET THE RESULT
066740  174010                      STD     AC0,(R0)
066742  012701  067404              MOV     #BBP10,R1       ;IS IT CORRECT
066746  012702  000004              MOV     #4,R2
066752  022021              BB10:   CMP     (R0)+,(R1)+
066754  001401                      BEQ     BB13
066756  104000                      EMT                     ;
066760  077204              BB13:   SOB     R2,BB10
066762  020405                      CMP     R4,R5           ;FPS CORRECT?
066764  001401                      BEQ     BB14
066766  104000                      EMT                     ;
                            ;EXPONENT DIFFERENCE=25=31 (OCT) FD=0
066770  012700  067304      BB14:   MOV     #BBPAT0,R0      ;SET UP AC0 OPERAND
066774  172410                      LDD     (R0),AC0
066776  012704  003000              MOV     #3000,R4        ;SET FIV AND FIV
                                                            ;CLEAR FD
067002  170104                      LDFPS   R4
067004  012700  067314              MOV     #BBPAT1,R0      ;FSRC
067010  172010              BB15:   ADDF    (R0),AC0        ;TEST INSTRUCTION
067012  170205                      STFPS   R5
067014  170011                      SETD                    ;REENTED DOUBLE MODE.
067016  012700  067274              MOV     #BBDAT0,R0      ;GET THE RESULT
067022  174010                      STD     AC0,(R0)
067024  012701  067304              MOV     #BBPAT0,R1      ;IS THE RESULT
067030  012702  000002              MOV     #2,R2           ;CORRECT?
067034  022021              BB16:   CMP     (R0)+,(R1)+
067036  001401                      BEQ     BB17
067040  104000                      EMT                     ;
067042  077204              BB17:   SOB     R2,BB16
```

```
067044  020405                          CMP     R4,R5           ;IS FPS CORRECT?
067046  001401                          BEQ     BB20
067050  104000                          EMT                     ;
                                ;EXPONENT DIFFERENCE=24=30 (OCT)
067052  012700  067334          BB20:   MOV     #BBPAT3,R0      ;SET UP AC0 OPERAND.
067056  172410                          LDD     (R0),AC0
067060  012704  003000                  MOV     #3000,R4        ;SET FIU,FIV. CLEAR FD.
067064  170104                          LDFPS   R4
067066  012700  067314                  MOV     #BBPAT1,R0      ;FSRC
067072  172010                  BB21:   ADDF    (R0),AC0        ;TEST INSTRUCTION
067074  170205                          STFPS   R5
067076  170011                          SETD                    ;REENTER DOUBLE MODE
067100  012700  067274                  MOV     #BBDAT0,R0      ;GET THE RESULT
067104  174010                          STD     AC0,(R0)
067106  012701  067374                  MOV     #BBP7,R1        ;IS THE RESULT CORRECT?
067112  012702  000002                  MOV     #2,R2
067116  022021                  BB22:   CMP     (R0)+,(R1)+
067120  001401                          BEQ     BB25
067122  104000                          EMT                     ;
067124  077204                  BB25:   SOB     R2,BB22
067126  020405                          CMP     R4,R5
067130  001401                          BEQ     BB26
067132  104000                          EMT                     ;
                                ;EXPONENT DIFFERENCE=1
067134  012704  003200          BB26:   MOV     #3200,R4
067140  170104                          LDFPS   R4              ;SET UP AC0 OPERAND
067142  012700  067354                  MOV     #BBPAT5,R0
067146  172410                          LDD     (R0),AC0
067150  012700  067314                  MOV     #BBPAT1,R0      ;FSRC
067154  172010                  BB27:   ADDD    (R0),AC0        ;TEST INSTRUCTION
067156  170205                          STFPS   R5
067160  012700  067274                  MOV     #BBDAT0,R0      ;GET THE RESULT.
067164  174010                          STD     AC0,(R0)
067166  012701  067414                  MOV     #BBP11,R1       ;IS IT CORRECT?
067172  012702  000004                  MOV     #4,R2
067176  022021                  BB30:   CMP     (R0)+,(R1)+
067200  001401                          BEQ     BB31
067202  104000                          EMT                     ;
067204  077204                  BB31:   SOB     R2,BB30
067206  020405                          CMP     R4,R5           ;IS FPS CORRECT
067210  001401                          BEQ     BB32
067212  104000                          EMT                     ;
                                ;EXPONENT DIFFERENCE=100=144 (OCT)
067214  012704  003200          BB32:   MOV     #3200,R4
067220  170104                          LDFPS   R4              ;SET FIV,FIV AND FD
067222  012700  067364                  MOV     #BBPAT6,R0      ;SET UP AC0 OPERAND.
067226  172410                          LDD     (R0),AC0
067230  012700  067314                  MOV     #BBPAT1,R0      ;FSRC
067234  172010                  BB33:   ADDD    (R0),AC0        ;TEST INSTRUCTION
067236  170205                          STFPS   R5
067240  012700  067274                  MOV     #BBDAT0,R0      ;GET THE RESULT
067244  174010                          STD     AC0,(R0)
067246  012701  067364                  MOV     #BBPAT6,R1      ;IS IT CORRECT
067252  012702  000004                  MOV     #4,R2
067256  022021                  BB34:   CMP     (R0)+,(R1)+
067260  001401                          BEQ     BB35
```

```
067262  104000                  EMT                     ;
067264  077204          BB35:   SOB     R2,BB34
067266  020405                  CMP     R4,R5           ;IS FPS CORRECT
067270  001455                  BEQ     BBDONE
067272  104000                  EMT                     ;
067274  000000          BBDAT0: 0
067276  000000                  0
067300  000000                  0
067302  000000                  0
067304  006400          BBPAT0: 6400                    ;F(AC)=E(FSRC)+25=26
067306  000000                  0                       ;                 =32(OCT)
067310  000000                  0
067312  000000                  0
067314  000200          BBPAT1: 200
067316  000000                  0                       ;E(FSRC)=1
067320  000000                  0
067322  000000                  0
067324  016400          BBPAT2: 16400
067326  000000                  0                       ;E(AC)=E(FSRC)+57=58
067330  000000                  0                       ;                 =72(OCT)
067332  000000                  0
067334  006200          BBPAT3: 6200                    ;E(AC)=E(FSRC)+24=25
067336  000000                  0                       ;                 =31(OCT)
067340  000000                  0
067342  000000                  0
067344  016200          BBPAT4: 16200                   ;E(AC)=E(FSR6)+56=57
067346  000000                  0                       ;                 =71(OCT)
067350  000000                  0
067352  000000                  0
067354  000400          BBPAT5: 400                     ;E(AC)=E(FSRC)+1=2
067356  000000                  0
067360  000000                  0
067362  000000                  0
067364  031200          BBPAT6: 31200                   ;E(AC)=E(FSRC)+100=101
067366  000000                  0                       ;                  =145(OCT)
067370  000000                  0
067372  000000                  0
067374  006200          BBP7:   6200                    ;BBPAT3 RES
067376  000001                  1
067400  000000                  0
067402  000000                  0
067404  016200          BBP10:  16200                   ;BBPAT4 RES
067406  000000                  0
067410  000000                  0
067412  000001                  1
067414  000500          BBP11:  500                     ;BBPAT5 RES
067416  000000                  0
067420  000000                  0
067422  000000                  0
067424                  BBDONE:
067424  004767  035142          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                        ;THE USER TYPED CONTROL G?).
                        ;**********************************************************************************
```

```
                                  ;TEST 465       ADDD WITH NEGATVE OPRANDS TEST
                                  ;*******************************************************************************
067430                            TS465:
                                  ;BOTH OPERANDS NEGATIVE
067430  012704  003200                    MOV     #3200,R4        ;SET FIO, FIV, AND FD
067434  170104                            LDFPS   R4
067436  012700  070214                    MOV     #DDP1,R0                ;SET AC0 OPERAND
067442  172410                            LDD     (R0),AC0
067444  012700  070214                    MOV     #DDP1,R0                ;ESRC
067450  172010                    DD2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
067452  170205                            STFPS   R5                      ;GET FPS
067454  012700  070174                    MOV     #DDDATO,R0              ;GET THE RESULT
067460  174010                            STD     AC0,(R0)
067462  012701  070314                    MOV     #DDP9,R1                ;IS IT CORRECT
067466  012702  000004                    MOV     #4,R2
067472  022021                    DD3:    CMP     (R0)+,(R1)+
067474  001401                            BEQ     DD6
067476  104000                            EMT                     ;
067500  077204                    DD6:    SOB     R2,DD3
067502  052704  000010                    BIS     #10,R4
067506  020405                            CMP     R4,R5                   ;FPS CORRECT?
067510  001401                            BEQ     DD7
067512  104000                            EMT                     ;
                                  ;AC POS FSRC NEG          AC=-FSRC
067514  012704  003200            DD7:    MOV     #3200,R4                ;SET FIO, FIV, AND FD
067520  170104                            LDFPS   R4
067522  012700  070224                    MOV     #DDP2,R0                ;SET AC0 OPERAND
067526  172410                            LDD     (R0),AC0
067530  012700  070214                    MOV     #DDP1,R0                ;FSPC
067534  172010                    DD8:    ADDD    (R0),AC0                ;TEST INSTRUCTION
067536  170205                            STFPS   R5                      ;GET FPS
067540  012700  070174                    MOV     #DDDATO,R0              ;GET THE RESULT
067544  174010                            STD     AC0,(R0)
067546  012701  070204                    MOV     #DDP0,R1                ;IS IT CORRECT
067552  012702  000004                    MOV     #4,R2
067556  022021                    DD10:   CMP     (R0)+,(R1)+
067560  001401                            BEQ     DD11
067562  104000                            EMT                     ;
067564  077204                    DD11:   SOB     R2,DD10
067566  052704  000004                    BIS     #4,R4
067572  020405                            CMP     R4,R5                   ;FPS CORRECT?
067574  001401                            BEQ     DD12
067576  104000                            EMT                     ;
                                  ;AC NEG FSRC POS                AC=-FSRC
067600  012704  003200            DD12:   MOV     #3200,R4                ;SET FIU, FIV, AND FD
067604  170104                            LDFPS   R4
067606  012700  070214                    MOV     #DDP1,R0                ;SET AC0 OPERAND
067612  172410                            LDD     (R0),AC0
067614  012700  070224                    MOV     #DDP2,R0                ;FSRC
067620  172010                    DD13:   ADDD    (R0),AC0                ;TEST INSTRUCTION
067622  170205                            STFPS   R5                      ;GET FPS
067624  012700  070174                    MOV     #DDDATO,R0              ;GET THE RESULT
067630  174010                            STD     AC0,(R0)
067632  012701  070204                    MOV     #DDP0,R1                ;IS IT CORRECT
067636  012702  000004                    MOV     #4,R2
067642  022021                    DD14:   CMP     (R0)+,(R1)+
```

```
067644  001401                          BEQ     DD15
067646  104000                          EMT                     ;
067650  077204                  DD15:   SOB     R2,DD14
067652  052704  000004                  BIS     #4,R4
067656  020405                          CMP     R4,R5                   ;EPS CORRECT?
067660  001401                          BEQ     DD16
067662  104000                          EMT                     ;
                                ;ACO POC        FSRC    NEG     /AC/ > /FSRC/
067664  012704  003200          DD16:   MOV     #3200,R4                ;SET FIV, FIV AND FD
067670  170104                          LDFPS   R4
067672  012700  070234                  MOV     #DDP3,R0                ;SET ACO OPERAND
067676  172410                          LDD     (R0),AC0
067700  012700  070264                  MOV     #DDP6,R0                ;ESPC
067704  172010                  DD17:   ADDD    (R0),AC0                ;TEST INSTRUCTION
067706  170205                          STFPS   R5                      ;GET FPS
067710  012700  070174                  MOV     #DDDAT0,R0              ;GET THE RESULT
067714  174010                          STD     AC0,(R0)
067716  012701  070274                  MOV     #DDP7,R1                ;IS IT CORRECT
067722  012702  000004                  MOV     #4,R2
067726  022021                  DD18:   CMP     (R0)+,(R1)+
067730  001401                          BEQ     DD21
067732  104000                          EMT                     ;
067734  077204                  DD21:   SOB     R2,DD18
067736  020405                          CMP     R4,R5                   ;EPS CORRECT?
067740  001401                          BEQ     DD22
067742  104000                          EMT                     ;
                                ;AC NEG FSRC    POS     /FSRC/ > /AC/
067744  012704  003200          DD22:   MOV     #3200,R4                ;SET FIO,FIV, AND FD
067750  170104                          LDFPS   R4
067752  012700  070264                  MOV     #DDP6,R0                ;SET ACO OPERAND
067756  172410                          LDD     (R0),AC0
067760  012700  070234                  MOV     #DDP3,R0                ;FSPC
067764  172010                  DD23:   ADDD    (R0),AC0                ;TEST INSTRUCTION
067766  170205                          STFPS   R5                      ;GET FPS
067770  012700  070174                  MOV     #DDDAT0,R0              ;GET THE RESULT
067774  174010                          STD     AC0,(R0)
067776  012701  070274                  MOV     #DDP7,R1                ;IS IT CORRECT?
070002  012702  000004                  MOV     #4,R2
070006  022021                  DD24:   CMP     (R0)+,(R1)+
070010  001401                          BEQ     DD27
070012  104000                          EMT                     ;
070014  077204                  DD27:   SOB     R2,DD24
070016  020405                          CMP     R4,R5                   ;FPS CORRECT?
070020  001401                          BEQ     DD30
070022  104000                          EMT                     ;
                                ;ACO POS        FSRC    NEG     /AC/</FRSRC/
070024  012704  003200          DD30:   MOV     #3200,R4                ;SET FIO,FIV,AND FD
070030  170104                          LDFPS   R4
070032  012700  070244                  MOV     #DDP4,R0                ;SET ACO OPERAND
070036  172410                          LDD     (R0),AC0
070040  012700  070254                  MOV     #DDP5,R0                ;FSPC
070044  172010                  DD31:   ADDD    (R0),AC0                ;TEST INSTRUCTION
070046  170205                          STFPS   R5                      ;GET FPS
070050  012700  070174                  MOV     #DDDAT0,R0              ;GET THE RESULT
070054  174010                          STD     AC0,(R0)
070056  012701  070304                  MOV     #DDP8,R1                ;IS IT CORRECT
```

```
070062 012702 000004            MOV     #4,R2
070066 022021           DD32:   CMP     (R0)+,(R1)+
070070 001401                   BEQ     DD35
070072 104000                   EMT                     ;
070074 077204           DD35:   SOB     R2,DD32
070076 052704 000010            BIS     #10,R4
070102 020405                   CMP     R4,R5                           ;FPS CORRECT?
070104 001401                   BEQ     DD36
070106 104000                   EMT                     ;
                        ;AC0 NEG        FSRC    POS     /FSRC/</AC/
070110 012704 003200    DD36:   MOV     #3200,R4                        ;SET FIO, FIV, AND FD
070114 170104                   LDFPS   R4
070116 012700 070254            MOV     #DDP5,R0                        ;SET AC0 OPERAND
070122 172410                   LDD     (R0),AC0
070124 012700 070244            MOV     #DDP4,R0                        ;FSPC
070130 172010           DD37:   ADDD    (R0),AC0                        ;TEST INSTRUCTION
070132 170205                   STFPS   R5                              ;GET FPS
070134 012700 070174            MOV     #DDDATO,R0                      ;GET THE RESULT
070140 174010                   STD     AC0,(R0)
070142 012701 070304            MOV     #DDP8,R1                        ;IS IT CORRECT
070146 012702 000004            MOV     #4,R2
070152 022021           DD38:   CMP     (R0)+,(R1)+
070154 001401                   BEQ     DD41
070156 104000                   EMT                     ;
070160 077204           DD41:   SOB     R2,DD38
070162 052704 000010            BIS     #10,R4
070166 020405                   CMP     R4,R5                           ;FPS CORRECT?
070170 001455                   BEQ     DDDONE
070172 104000                   EMT                     ;
070174 000000           DDDATO: 0
070176 000000                   0
070200 000000                   0
070202 000000                   0
070204 000000           DDP0:   0
070206 000000                   0
070210 000000                   0
070212 000000                   0
070214 100200           DDP1:   100200                                  ;-DDP2
070216 000000                   0
070220 000000                   0
070222 000000                   0
070224 000200           DDP2:   200                                     ;-DDP1
070226 000000                   0
070230 000000                   0
070232 000000                   0
070234 001100           DDP3:   1100                                    ;EXP=4
070236 000000                   0                                       ;FRAC=...110...
070240 000000                   0
070242 000000                   0
070244 000600           DDP4:   600                                     ;EXP=3
070246 000000                   0                                       ;FRAC=...100...
070250 000000                   0
070252 000000                   0
070254 101100           DDP5:   101100                                  ;-DDP3
070256 000000                   0
070260 000000                   0
```

```
070262 000000                        0
070264 100600                DDP6:   100600                                 ;-DDP4
070266 000000                        0
070270 000000                        0
070272 000000                        0
070274 001000                DDP7:   1000                                   ;DDP3+DDP6
070276 000000                        0
070300 000000                        0
070302 000000                        0
070304 101000                DDP8:   101000                                 ;DDP5+DDP4
070306 000000                        0
070310 000000                        0
070312 000000                        0
070314 100400                DDP9:   100400                                 ;DDP1+DDP1
070316 000000                        0
070320 000000                        0
070322 000000                        0
070324                       DDDONE:
070324 004767 034242                 JSR     PC,.RSET               ;GO INITIALIZE THE FPS AND STACK; AND
                                                                    ;SEE IF THE USER HAS EXPRESSED
                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                    ;THE USER TYPED CONTROL G?).
                             ;*************************************************************************
                             ;TEST 466        SUBD TEST
                             ;*************************************************************************
070330                       TS466:
                             ;       USE POSITIVE OPERANDS
070330 012704 003200                 MOV     #3200,R4                       ;SET FIU, FIV, AND FD
070334 170104                        LDFPS   R4
070336 012700 070522                 MOV     #EEP1,R0                       ;SET AC0 OPERAND
070342 172410                        LDD     (R0),AC0
070344 012700 070522                 MOV     #EEP1,R0                       ;FSPC
070350 173010                EE2:    SUBD    (R0),AC0                       ;TEST INSTRUCTION
070352 170205                        STFPS   R5                             ;GET FPS
070354 012700 070500                 MOV     #EEDAT0,R0                     ;GET THE RESULT
070360 174010                        STD     AC0,(R0)
070362 012701 070510                 MOV     #EEP0,R1                       ;IS IT CORRECT?
070366 012702 000004                 MOV     #4,R2
070372 022021                EE3:    CMP     (R0)+,(R1)+
070374 001401                        BEQ     EE6
070376 104000                        EMT                            ;
070400 077204                EE6:    SOB     R2,EE3
070402 052704 000004                 BIS     #4,R4
070406 020405                        CMP     R4,R5                          ;FPS CORRECT?
070410 001401                        BEQ     EE7
070412 104000                        EMT                            ;
                                     ;USE NEGATIVE OPERANDS
070414 012704 003200         EE7:    MOV     #3200,R4                       ;SET FIO, FIV, AND FD
070420 170104                        LDFPS   R4
070422 012700 070542                 MOV     #EEP3,R0                       ;SET AC0 OPERAND
070426 172410                        LDD     (R0),AC0
070430 012700 070542                 MOV     #EEP3,R0                       ;FSPC
070434 173010                EE8:    SUBD    (R0),AC0                       ;TEST INSTRUCTION
070436 170205                        STFPS   R5                             ;GET FPS
070440 012700 070500                 MOV     #EEDAT0,R0                     ;GET THE RESULT
```

```
070444  174010                          STD     AC0,(R0)
070446  012701  070510                  MOV     #EEP0,R1                ;IS IT CORRECT?
070452  012702  000004                  MOV     #4,R2
070456  022021                  EE9:    CMP     (R0)+,(R1)+
070460  001401                          BEQ     EE12
070462  104000                          EMT                             ;
070464  077204                  EE12:   SOB     R2,EE9
070466  052704  000004                  BIS     #4,R4
070472  020405                          CMP     R4,R5                   ;FPS CORRECT?
070474  001432                          BEQ     EEDONE
070476  104000                          EMT                             ;
070500  000000                  EEDATO: 0
070502  000000                          0
070504  000000                          0
070506  000000                          0
070510  000000                  EEP0:   0
070512  000000                          0
070514  000000                  00000
070516  000000                          0
070520  000000                          0
070522  000200                  EEP1:   200
070524  000000                          0
070526  000000                          0
070530  000000                          0
070532  000400                  EEP2:   400
070534  000000                          0
070536  000000                          0
070540  000000                          0
070542  100200                  EEP3:   100200
070544  000000                          0
070546  000000                          0
070550  000000                          0
070552  100400                  EEP4:   100400
070554  000000                          0
070556  000000                          0
070560  000000                          0
070562                          EEDONE:
070562  004767  034004                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).

                                ;*************************************************************************
                                ;TEST 467       NORMALIZE ALGORITHM TEST
                                ;*************************************************************************
070566                          TS467:
                                ;USE DATA PATTERNS THAT REQUIRE ONLY ONE LEFT SHIFT TO NORMALIZE
070566  012704  003200                  MOV     #3200,R4                ;SET FIO, FIV, AND FD
070572  170104                          LDFPS   R4
070574  012700  070756                  MOV     #FFP2,R0                ;SET AC0 OPERAND
070600  172410                          LDD     (R0),AC0
070602  012700  070766                  MOV     #FFP3,R0                ;FSPC
070606  172010                  FF2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
070610  170205                          STFPS   R5                      ;GET FPS
070612  012700  070726                  MOV     #FFDATO,R0              ;GET THE RESULT
070616  174010                          STD     AC0,(R0)
```

```
15360  070620  012701  070776           MOV     #FFP4, R1               ;IS IT CORRECT
```

```
070624  012702  000004          MOV     #4,R2
070630  022021          FF3:    CMP     (R0)+,(R1)+
070632  001401                  BEQ     FF4
070634  104000                  EMT                     ;
070636  077204          FF4:    SOB     R2,FF3
070640  020405                  CMP     R4,R5                   ;FPS CORRECT?
070642  001401                  BEQ     FF5
070644  104000                  EMT                     ;
                        ;USE DATA PATTERNS WHICH REQUIRE 56 LEFT SHIFTS TO NORMALIZE
                        ;THE RESULT
070646  012704  003200  FF5:    MOV     #3200,R4                ;SET FIU, FIV, AND FD
070652  170104                  LDFPS   R4
070654  012700  070736          MOV     #FFP0,R0                ;SET AC0 OPERAND
070660  172410                  LDD     (R0),AC0
070662  012700  070746          MOV     #FFP1,R0                ;FSRC
070666  172010          FF6:    ADDD    (R0),AC0                ;TEST INSTRUCTION
070670  170205                  STFPS   R5                      ;GET FPS
070672  012700  070726          MOV     #FFDATO,R0              ;GET THE RESULT
070676  174010                  STD     AC0,(R0)
070700  012701  070776          MOV     #FFP4,R1                ;IS IT CORRECT
070704  012702  000004          MOV     #4,R2
070710  022021          FF7:    CMP     (R0)+,(R1)+
070712  001401                  BEQ     FF10
070714  104000                  EMT                     ;
070716  077204          FF10:   SOB     R2,FF7
070720  020405                  CMP     R4,R5                   ;FPS CORRECT?
070722  001431                  BEQ     FFDONE
070724  104000                  EMT                     ;


070726  000000          FFDATO: 0
070730  000000                  0
070732  000000                  0
070734  000000                  0

070736  016000          FFP0:   16000
070740  000000                  0
070742  000000                  0
070744  000001                  1
070746  116000          FFP1:   116000
070750  000000                  0
070752  000000                  0
070754  000000                  0
070756  000500          FFP2:   500
070760  000000                  0
070762  000000                  0
070764  000000                  0
070766  100400          FFP3:   100400
070770  000000                  0
070772  000000                  0
070774  000000                  0
070776  000200          FFP4:   200                             ;FFP4=FFP0+FFP1
071000  000000                  0                               ;     =FFP3+FFP4
071002  000000                  0
071004  000000                  0
```

```
071006                      FFDONE:
071006  004767  033560              JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                            ;SEE IF THE USER HAS EXPRESSED
                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                            ;THE USER TYPED CONTROL G?).



                            ;*******************************************************************
                            ;*******************************************************************
                            ;FLOATING POINT SECOND PART
                            ;*******************************************************************
                            ;*******************************************************************


                            ;*************************************************************************************
                            ;TEST 470       ROUND\TRUNK TEST
                            ;*************************************************************************************
071012                      TS470:

                                    ;ROUND AND NORMALIZE TEST

071012  012704  003200              MOV     #3200,R4                ;SET FIU, FIV, AND FD
071016  170104                      LDFPS   R4
071020  012700  071422              MOV     #HHP0,R0                ;SET AC0 OPERAND
071024  172410                      LDD     (R0),AC0
071026  012700  071432              MOV     #HHP1,R0                ;FSPC
071032  172010              HH2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
071034  170205                      STFPS   R5                      ;GET FPS
071036  012700  071412              MOV     #HHDAT0,R0              ;GET THE RESULT
071042  174010                      STD     AC0,(R0)
071044  012701  071442              MOV     #HHP2,R1                ;IS IT CORRECT
071050  012702  000004              MOV     #4,R2
071054  022021              HH3:    CMP     (R0)+,(R1)+
071056  001401                      BEQ     HH6
071060  104000                      EMT                     ;
071062  077204              HH6:    SOB     R2,HH3
071064  020405                      CMP     R4,R5           ;FPS CORRECT?
071066  001401                      BEQ     HH7
071070  104000                      EMT                     ;

                            ;THIS IS A TEST OF THE ABILITY
                            ;OF NORMALIZE TO PRODUCE A ZERO EXP. AND
                            ;OF THE R\T ALGORITHM TO PROPERLY SET THE FPS

071072  012704  043200      HH7:    MOV     #043200,R4              ;SET FIU,FIV,AND FD
                                                                    ;FID
071076  170104                              LDFPS   R4
071100  012700  071472              MOV     #HHP5,R0                ;SET AC0 OPERAND
071104  172410                      LDD     (R0),AC0
071106  012700  071502              MOV     #HHP6,R0                ;FSPC
071112  172010              HH8:    ADDD    (R0),AC0                ;TEST INSTRUCTION
071114  170205                      STFPS   R5                      ;GET FPS
071116  012700  071412              MOV     #HHDAT0,R0              ;GET THE RESULT
071122  174010                      STD     AC0,(R0)
```

```
071124  012701  071462          MOV     #HHP4,R1                ;IS IT CORRECT
071130  012702  000004          MOV     #4,R2
071134  022021          HH9:    CMP     (R0)+,(R1)+
071136  001401                  BEQ     HH10
071140  104000                  EMT                     ;
071142  077204          HH10:   SOB     R2,HH9
071144  052704  100004          BIS     #100004,R4
071150  020405                  CMP     R4,R5
071152  001401                  BEQ     HH11
071154  104000                  EMT                     ;
                        ;THIS IS A TEST OF THE R\T ALGORITHN'S
                        ;ABILITY TO SET BOTH N AND Z ON A - 0 RESULT
071156  012704  043200  HH11:   MOV     #043200,R4              ;SET FIV, FIV, AND FD
071162  170104                  LDFPS   R4
071164  012700  071522          MOV     #HHP8,R0                ;SET AC0 OPERAND
071170  172410                  LDD     (R0),AC0
071172  012700  071532          MOV     #HHP9,R0                ;FSPC
071176  172010          HH12:   ADDD    (R0),AC0                ;TEST INSTRUCTION
071200  170205                  STFPS   R5                      ;GET FPS
071202  012700  071412          MOV     #HHDAT0,R0              ;GET THE RESULT
071206  174010                  STD     AC0,(R0)
071210  012701  071512          MOV     #HHP7,R1                ;IS IT CORRECT
071214  012702  000004          MOV     #4,R2
071220  022021          HH13:   CMP     (R0)+,(R1)+
071222  001401                  BEQ     HH16
071224  104000                  EMT                     ;
071226  077204          HH16:   SOB     R2,HH13
071230  052704  100014          BIS     #100014,R4              ;FPS CORRECT?
071234  020405                  CMP     R4,R5
071236  001401                  BEQ     HH17
071240  104000                  EMT                     ;
                        ;TEST THAT CC ARE CLEARED BY R\T
071242  012704  000200  HH17:   MOV     #00200,R4               ;SET FIV, FIV, AND FD
071246  170104                  LDFPS   R4
071250  012700  071522          MOV     #HHP8,R0                ;SET AC0 OPERAND
071254  172410                  LDD     (R0),AC0
071256  012700  071522          MOV     #HHP8,R0                ;FSPC
071262  172010          HH18:   ADDD    (R0),AC0                ;TEST INSTRUCTION
071264  170205                  STFPS   R5                      ;GET FPS
071266  012700  071412          MOV     #HHDAT0,R0              ;GET THE RESULT
071272  174010                  STD     AC0,(R0)
071274  012701  071542          MOV     #HHP10,R1               ;IS IT CORRECT
071300  012702  000004          MOV     #4,R2
071304  022021          HH19:   CMP     (R0)+,(R1)+
071306  001401                  BEQ     HH20
071310  104000                  EMT                     ;
071312  077204          HH20:   SOB     R2,HH19
071314  052704  000000          BIS     #00000,R4               ;FPS CORRECT?
071320  020405                  CMP     R4,R5
071322  001401                  BEQ     HH21
071324  104000                  EMT                     ;
                        ;TEST THAT N IS SET BY R\T
071326  012704  003200  HH21:   MOV     #3200,R4                ;SET FIV, FIV, AND FD
071332  170104                  LDFPS   R4
```

```
071334 012700 071472          MOV     #HHP5,R0          ;SET AC0 OPERAND
071340 172410                 LDD     (R0),AC0
071342 012700 071472          MOV     #HHP5,R0          ;FSPC
071346 172010         HH22:   ADDD    (R0),AC0          ;TEST INSTRUCTION
071350 170205                 STFPS   R5                ;GET FPS
071352 012700 071412          MOV     #HHDAT0,R0        ;GET THE RESULT
071356 174010                 STD     AC0,(R0)
071360 012701 071552          MOV     #HHP11,R1         ;IS IT CORRECT
071364 012702 000004          MOV     #4,R2
071370 022021         HH23:   CMP     (R0)+,(R1)+
071372 001401                 BEQ     HH24
071374 104000                 EMT                 ;
071376 077204         HH24:   SOB     R2,HH23
071400 052704 000010          BIS     #10,R4
071404 020405                 CMP     R4,R5             ;FPS CORRECT?
071406 001465                 BEQ     HHDONE
071410 104000                 EMT                 ;
071412 000000         HHDAT0: 0
071414 000000                 0
071416 000000                 0
071420 000000                 0
071422 000452         HHP0:   452
071424 125252                 125252
071426 125252                 125252
071430 125253                 125253
071432 000252         HHP1:   252
071434 125252                 125252
071436 125252                 125252
071440 125252                 125252
071442 000600         HHP2:   600                 ;HHP0 + HHP1 WITH
071444 000000                 0                   ;PROPER NORMALIZATION
071446 000000                 0
071450 000000                 0
071452 000400         HHP3:   400                 ;HHP0 + HHP1 WITH
071454 000000                 0                   ;BAD NORMALIZATION
071456 000000                 0
071460 000000                 0
071462 000000         HHP4:   0
071464 000000                 0
071466 000000                 0
071470 000000                 0
071472 100200         HHP5:   100200
071474 000000                 0
071476 000000                 0
071500 000000                 0
071502 000300         HHP6:   300
071504 000000                 0
071506 000000                 0
071510 000000                 0
071512 100000         HHP7:   100000              ;HHP7 = HHP8 + HHP9
071514 000000                 0                   ;     = HHP5 + HHP6
071516 000000                 0
071520 000000                 0
071522 000200         HHP8:   200
071524 000000                 0
071526 000000                 0
```

```
071530 000000                                0
071532 100300                        HHP9:   100300
071534 000000                                0
071536 000000                                0
071540 000000                                0
071542 000400                        HHP10:  400                     ;HHP10 = HHP8 + HHP8
071544 000000                                0
071546 000000                                0
071550 000000                                0
071552 100400                        HHP11:  100400                  ;HHP11 = HHP5 + HHP5
071554 000000                                0
071556 000000                                0
071560 000000                                0
071562                               HHDONE:
071562 004767 033004                         JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                     ;SEE IF THE USER HAS EXPRESSED
                                                                     ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                     ;THE USER TYPED CONTROL G?).


                                     ;*******************************************************************************
                                     ;TEST 471        OVER\UNDER TEST
                                     ;*******************************************************************************
071566                               TS471:

                                     ;TEST OVERFLOW CONDITION WITH TRAP DISABLER FIV=0
071566 012704 000200                         MOV     #200,R4         ;CLEAR FIU, FIV, AND SET FD
071572 170104                                LDFPS   R4
071574 012737 071712 000244                  MOV     #GGER0,@#FPVECT
071602 012700 072514                         MOV     #GGP5,R0        ;SET AC0 OPERAND
071606 172410                                LDD     (R0),AC0
071610 012700 072514                         MOV     #GGP5,R0        ;FSRC
071614 172010                        GG2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
071616 170205                                STFPS   R5              ;GET FPS
071620 012700 072444                         MOV     #GGDAT0,R0      ;GET THE RESULT
071624 174010                                STD     AC0,(R0)
071626 012701 072524                         MOV     #GGP6,R1        ;IS IT CORRECT
071632 012702 000004                         MOV     #4,R2
071636 022021                        GG3:    CMP     (R0)+,(R1)+
071640 001401                                BEQ     GG4
071642 104000                                EMT                     ;
071644 077204                        GG4:    SOB     R2,GG3
071646 052704 000006                         BIS     #6,R4           ;FPS CORRECT?
071652 020405                                CMP     R4,R5
071654 001401                                BEQ     GG5
071656 104000                                EMT                     ;
                                             ;TEST OVERFLOW WITH TRAPS ENABLED
                                             ;FIV = 1
071660 012704 001200                 GG5:    MOV     #1200,R4        ;CLEAR FIU, SET FIV, AND FD
071664 170104                                LDFPS   R4
071666 012737 071714 000244                  MOV     #GG7,@#FPVECT
071674 012700 072514                         MOV     #GGP5,R0        ;SET AC0 OPERAND
071700 172410                                LDD     (R0),AC0
071702 012700 072514                         MOV     #GGP5,R0        ;FSPC
071706 172010                        GG6:    ADDD    (R0),AC0        ;TEST INSTRUCTION
```

```
071710  170000                          CFCC                            ;NO OVERFLOW TRAP OCCURED
071712                          GGER0:
071712  104000                          EMT                             ;
071714  012703  071710          GG7:    MOV     #GG6+2,R3
071720  020316                          CMP     R3,(SP)                 ;CHECK STACK DATA
071722  001401                          BEQ     1$
071724  104000                          EMT                             ;
071726  022626                  1$:     CMP     (SP)+,(SP)+
071730  170205                          STFPS   R5
071732  012700  072444                  MOV     #GGDAT0,R0              ;GET THE RESULT
071736  174010                          STD     AC0,(R0)
071740  012701  072524                  MOV     #GGP6,R1                ;IS IT CORRECT
071744  012702  000004                  MOV     #4,R2
071750  022021                  GG8:    CMP     (R0)+,(R1)+
071752  001401                          BEQ     GG9
071754  104000                          EMT                             ;
071756  077204                  GG9:    SOB     R2,GG8
071760  052704  100006                  BIS     #100006,R4              ;EXACT ZERO RESULTED IF OVERFLOW
071764  020405                          CMP     R4,R5                   ;FPS CORRECT?, CHECK FER, FZ, FV
071766  001401                          BEQ     1$
071770  104000                          EMT                             ;
071772  012704  000010          1$:     MOV     #10,R4
                                ;CHECK FEC
071776  170305                          STST    R5
072000  020405                          CMP     R4,R5
072002  001401                          BEQ     GG10
072004  104000                          EMT                             ;
                                ;CHECK UNDER FLOW CONDITION WITH
                                ;TRAPS DISABLED (FIU = 0)
072006  012704  000200          GG10:   MOV     #0200,R4                ;SET FIU, FIV, AND FD
072012  170104                          LDFPS   R4
072014  012737  071712  000244          MOV     #GGER0,@#FPVECT
072022  012700  072464                  MOV     #GGP2,R0                ;SET AC0 OPERAND
072026  172410                          LDD     (R0),AC0                ;FSRC
072030  012700  072474                  MOV     #GGP3,R0
072034  172010                  GG11:   ADDD    (R0),AC0                ;TEST INSTRUCTION
072036  170205                          STFPS   R5                      ;GET FPS
072040  012700  072444                  MOV     #GGDAT0,R0              ;GET THE RESULT
072044  174010                          STD     AC0,(R0)
072046  012701  072524                  MOV     #GGP6,R1                ;IS IT CORRECT
072052  012702  000004                  MOV     #4,R2
072056  022021                  GG12:   CMP     (R0)+,(R1)+
072060  001401                          BEQ     GG13
072062  104000                          EMT                             ;
072064  077204                  GG13:   SOB     R2,GG12
072066  052704  000004                  BIS     #4,R4                   ;FPS CORRECT?
072072  020405                          CMP     R4,R5
072074  001401                          BEQ     GG14
072076  104000                          EMT                             ;
                                        ;CHECK UNDERFLOW CONDITION WITH
                                        ;TRAP ENABLED (FIU = 1)
072100  012704  002200          GG14:   MOV     #2200,R4                ;SET FIU, FIV, AND FD
072104  170104                          LDFPS   R4
072106  012737  072132  000244          MOV     #GG16,@#FPVECT
072114  012700  072464                  MOV     #GGP2,R0                ;SET AC0 OPERAND
072120  172410                          LDD     (R0),AC0                ;FSPC
```

```
072122 012700 072474                  MOV    #GGP3,R0
072126 172010                 GG15:   ADDD   (R0),AC0        ;TEST INSTRUCTION
072130 170000                         CFCC
072132 012703 072130          GG16:   MOV    #GG15+2,R3
072136 021603                         CMP    (SP),R3
072140 001401                         BEQ    1$
072142 104000                         EMT                    ;
072144 022626                 1$:     CMP    (SP)+,(SP)+
072146 170205                         STFPS  R5              ;GET FPS
072150 012700 072444                  MOV    #GGDAT0,R0      ;GET THE RESULT
072154 174010                         STD    AC0,(R0)
072156 012701 072534                  MOV    #GGP7,R1        ;IS IT CORRECT
072162 012702 000004                  MOV    #4,R2
072166 022021                 GG17:   CMP    (R0)+,(R1)+
072170 001401                         BEQ    GG18
072172 104000                         EMT                    ;
072174 077204                 GG18:   SOB    R2,GG17
072176 052704 100000                  BIS    #100000,R4
072202 020405                         CMP    R4,R5           ;FPS CORRECT?
072204 001401                         BEQ    1$
072206 104000                         EMT                    ;
072210 012704 000012          1$:     MOV    #12,R4
                              ;CHECK FEC
072214 170305                         STST   R5
072216 020405                         CMP    R4,R5
072220 001401                         BEQ    GG19
072222 104000                         EMT                    ;
                              ;CHECK UNDERFLOW CONDITION WITH TRAPS
                              ;DISABLED (FIU = 0)
072224 012704 000200          GG19:   MOV    #0200,R4        ;SET FIU, FIV, AND FD
072230 170104                         LDFPS  R4
072232 012737 072350 000244           MOV    #GGER14,@#FPVECT
072240 012700 072464                  MOV    #GGP2,R0        ;SET AC0 OPERAND
072244 172410                         LDD    (R0),AC0
072246 012700 072544                  MOV    #GGP8,R0        ;FSPC
072252 172010                 GG20:   ADDD   (R0),AC0        ;TEST INSTRUCTION
072254 170205                         STFPS  R5              ;GET FPS
072256 012700 072444                  MOV    #GGDAT0,R0      ;GET THE RESULT
072262 174010                         STD    AC0,(R0)
072264 012701 072524                  MOV    #GGP6,R1        ;IS IT CORRECT
072270 012702 000004                  MOV    #4,R2
072274 022021                 GG21:   CMP    (R0)+,(R1)+
072276 001401                         BEQ    GG22
072300 104000                         EMT                    ;
072302 077204                 GG22:   SOB    R2,GG21
072304 052704 000004                  BIS    #4,R4           ;FPS CORRECT?
072310 020405                         CMP    R4,R5
072312 001401                         BEQ    GG23
072314 104000                         EMT                    ;
                                      ;CHECK UNDERFLOW CONDITION WITH TRAP
                                      ;ENABLED (FIU = 1)
072316 012704 002200          GG23:   MOV    #2200,R4        ;SET FIU, FIV, AND FD
072322 170104                         LDFPS  R4
072324 012737 072352 000244           MOV    #GG25,@#FPVECT
072332 012700 072464                  MOV    #GGP2,R0        ;SET AC0 OPERAND
072336 172410                         LDD    (R0),AC0
```

```
072340 012700 072544           MOV     #GGP8,R0          ;FSRC
072344 172010          GG24:   ADDD    (R0),AC0          ;TEST INSTRUCTION
072346 170000                  CFCC
072350                 GGER14:
072350 104000                  EMT                       ;
072352 012703 072346   GG25:   MOV     #GG24+2,R3
072356 020316                  CMP     R3,(SP)
072360 001401                  BEQ     1$
072362 104000                  EMT                       ;
072364 022626          1$:     CMP     (SP)+,(SP)+
072366 170205                  STFPS   R5                ;GET FPS
072370 012700 072444           MOV     #GGDATO,R0        ;GET THE RESULT
072374 174010                  STD     AC0,(R0)
072376 012701 072554           MOV     #GGP9,R1          ;IS IT CORRECT
072402 012702 000004           MOV     #4,R2
072406 022021          GG26:   CMP     (R0)+,(R1)+
072410 001401                  BEQ     GG27
072412 104000                  EMT                       ;
072414 077204          GG27:   SOB     R2,GG26
072416 052704 100004           BIS     #100004,R4
072422 020405                  CMP     R4,R5             ;FPS CORRECT?
072424 001401                  BEQ     1$
072426 104000                  EMT                       ;
072430 012704 000012   1$:     MOV     #12,R4
                               ;CHECK FEC
072434 170305                  STST    R5
072436 020405                  CMP     R4,R5
072440 001451                  BEQ     GGDONE
072442 104000                  EMT                       ;
072444 000000          GGDATO: 0
072446 000000                  0
072450 000000                  0
072452 000000                  0

072454 000300          GGP1:   300
072456 000000                  0
072460 000000                  0
072462 000000                  0
072464 100200          GGP2:   100200
072466 000000                  0
072470 000000                  0
072472 000000                  0
072474 000200          GGP3:   200
072476 000000                  0
072500 000000                  0
072502 000001                  1
072504 010200          GGP4:   10200
072506 000000                  0
072510 000000                  0
072512 000000                  0
072514 077600          GGP5:   77600                     ;OVER FLOW = GGP5 + GGP5
072516 000000                  0
072520 000000                  0
072522 000000                  0
072524 000000          GGP6:   0                         ;OVERFLOW RESULT
072526 000000                  0                         ;UNDERFLOW RESULT
```

```
072530  000000                          0                               ;GGP6 = GGP4 + GGP5
072532  000000                          0                               ;     = GGP3 + GGP2 (FIU = 0)
                                                                        ;     = GGP3 + GGP1
072534  062400                  GGP7:   62400                           ;GGP7 = GGP3 + GGP2 (FIU = 1)
072536  000000                          0
072540  000000                          0
072542  000000                          0
072544  000340                  GGP8:   340
072546  000000                          0
072550  000000                          0
072552  000000                          0
072554  000100                  GGP9:   100
072556  000000                          0
072560  000000                          0
072562  000000                          0
072564                          GGDONE:
072564  004767  032002                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).


                                ;*************************************************************************
                                ;TEST 472               LDCFD AND LDCDF TEST
                                ;*************************************************************************
072570                          TS472:
                                ;TEST FOR CORRECT AUTO INCREMENT CONSTANT.
072570  012704  000200                  MOV     #200,R4                 ;SET LONG INTEGER MODE
072574  170104                          LDFPS   R4
072576  012700  073326                  MOV     #HXP1,R0
072602  172410                          LDD     (R0),AC0
072604  012700  073336                  MOV     #HXP2,R0
072610  177420                  HX2:    LDCFD   (R0)+,AC0
072612  020027  073342                  CMP     R0,#HXP2+4              ;IS R0 CORRECT
072616  001401                          BEQ     HX3
072620  104000                          EMT                             ;
072622                          HX3:
072622  170205                          STFPS   R5                      ;GET FPS
072624  012700  073316                  MOV     #HXDAT0,R0
072630  174010                          STD     AC0,(R0)                ;GET AC0
072632  012701  073406                  MOV     #HXP7,R1                ;SEE IF RESULT IS
072636  012702  000004                  MOV     #4,R2                   ;CORRECT
072642  022120                  HX4:    CMP     (R1)+,(R0)+
072644  001401                          BEQ     HX7
072646  104000                          EMT                             ;
072650  077204                  HX7:    SOB     R2,HX4
072652  012704  000200                  MOV     #200,R4                 ;FPS CORRECT?
072656  020405                          CMP     R4,R5
072660  001401                          BEQ     HX8
072662  104000                          EMT                             ;
                                ;NOW    TEST    LDCDF
072664  012704  000200          HX8:    MOV #200,R4
072670  170104                          LDFPS R4

072672  012700  073326                  MOV     #HXP1,R0
```

```
072676  172410                          LDD     (R0),AC0

072700  012700  073336                  MOV     #HXP2,R0
072704  170001                          SETF
072706  177420                  HX9:    LDCDF   (R0)+,AC0       ;TEST INSTRUCTION
072710  020027  073346                  CMP     R0,#HXP2+10     ;WAS A GUOD
072714  001401                          BEQ     HX10
072716  104000                          EMT                     ;

072720                          HX10:
072720  170205                          STFPS   R5
072722  012700  073316                  MOV     #HXDAT0,R0
072726  170011                          SETD
072730  174010                          STD     AC0,(R0)        ;GET RESULT
072732  012701  073416                  MOV     #HXP8,R1
072736  012702  000004                  MOV     #4,R2
072742  022120                  HX11:   CMP     (R1)+,(R0)+     ;IS IT CORRECT?
072744  001401                          BEQ     HX14
072746  104000                          EMT                     ;
072750  077204                  HX14:   SOB     R2,HX11

072752  012704  000000                  MOV     #0,R4           ;FPS CORRECT?
072756  020405                          CMP     R4,R5
072760  001401                          BEQ     HX15
072762  104000                          EMT                     ;
                                ;TEST GR7 IMMEDIATE MODE CONSTANT
072764  012704  000200          HX15:   MOV     #200,R4
072770  170104                          LDFPS   R4              ;SET FD
072772  012737  073022  000004          MOV     #HXER9,@#ERRVECT
073000  005001                          CLR     R1
073002  177427  043243          HX16:   LDCFD   #5201,AC0
073006  005201                  HX165:  INC     R1
073010  005201                          INC     R1
073012  005201                          INC     R1
073014  020127  000003                  CMP     R1,#3           ;SEE IF PC WAS
073020  001401                          BEQ     HX17
073022                          HXER9:
073022  104000                          EMT                     ;
073024  012704  000200          HX17:   MOV     #200,R4
073030  170104                          LDFPS   R4
073032  012700  073376                  MOV     #HXP6,R0
073036  172410                          LDD     (R0),AC0
073040  012700  073336                  MOV     #HXP2,R0
073044  177410                  HX18:   LDCFD   (R0),AC0

073046  012700  073316                  MOV     #HXDAT0,R0
073052  174010                          STD     AC0,(R0)        ;GET RESULT.
073054  012701  073406                  MOV     #HXP7,R1
073060  012702  000004                  MOV     #4,R2
073064  022021                  HX19:   CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
073066  001401                          BEQ     HX20
073070  104000                          EMT                     ;
073072  077204                  HX20:   SOB     R2,HX19

                                ;TEST LDCFD WITH NEGATIVE OPERAND
073074  012704  000200                  MOV     #200,R4
```

```
073100  170104                          LDFPS   R4
073102  012700  073376                  MOV     #HXP6,R0
073106  172410                          LDD     (R0),AC0
073110  012700  073356                  MOV     #HXP4,R0
073114  177410                  HX22:   LDCFD   (R0),AC0

073116  012700  073316                  MOV     #HXDAT0,R0
073122  174010                          STD     AC0,(R0)                ;GET RESULT

073124  012701  073366                  MOV     #HXP5,R1
073130  012702  000004                  MOV     #4,R2
073134  022120                  HX23:   CMP     (R1)+,(R0)+
073136  001401                          BEQ     HX26
073140  104000                          EMT                             ;
073142  077204                  HX26:   SOB     R2,HX23

                                ;TEST   LDCFD   0

073144  012704  000200                  MOV     #200,R4
073150  170104                          LDFPS   R4

073152  012700  073326                  MOV     #HXP1,R0
073156  172410                          LDD     (R0),AC0
073160  172010                          ADDD    (R0),AC0

073162  012700  073326                  MOV     #HXP1,R0
073166  177410                  HX28:   LDCFD   (R0),AC0

073170  170205                          STFPS   R5

073172  012700  073316                  MOV     #HXDAT0,R0
073176  174010                          STD     AC0,(R0)                ;GET RESULT

073200  012701  073326                  MOV     #HXP1,R1
073204  012702  000004                  MOV     #4,R2
073210  022120                  HX29:   CMP     (R1)+,(R0)+             ;IS IT 0?
073212  001401                          BEQ     HX30
073214  104000                          EMT                             ;
073216  077204                  HX30:   SOB     R2,HX29

073220  012704  000204                  MOV     #204,R4                 ;FPS CORRECT
073224  020405                          CMP     R4,R5
073226  001401                          BEQ     HX31
073230  104000                          EMT                             ;
                                ;TEST   LDCFD   0
073232  012704  000200          HX31:   MOV     #200,R4
073236  170104                          LDFPS   R4
073240  012700  073376                  MOV     #HXP6,R0
073244  172410                          LDD     (R0),AC0
073246  012700  073326                  MOV     #HXP1,R0
073252  177410                  HX32:   LDCFD   (R0),AC0
073254  170205                          STFPS   R5
073256  012700  073316                  MOV     #HXDAT0,R0
073262  174010                          STD     AC0,(R0)                ;GET RESULT
073264  012701  073326                  MOV     #HXP1,R1
073270  012702  000004                  MOV     #4,R2
```

```
073274  022120                  HX33:   CMP     (R1)+,(R0)+     ;IS IT ZERO?
073276  001401                          BEQ     HX34
073300  104000                          EMT                     ;
073302  077204                  HX34:   SOB     R2,HX33

073304  012704  000204                  MOV     #204,R4         ;FPS CORRECT?
073310  020405                          CMP     R4,R5
073312  001445                          BEQ     HXDONE
073314  104000                          EMT                     ;

073316  000000                  HXDATO: 0
073320  000000                          0
073322  000000                          0
073324  000000                          0

073326  000000                  HXP1:   0
073330  000000                          0
073332  000000                          0
073334  000000                          0

073336  000577                  HXP2:   577
073340  177776                          177776
073342  177777                          177777
073344  177776                          177776
073346  005201                  HXP3:   5201
073350  000000                          0
073352  000000                          0
073354  000000                          0
073356  100577                  HXP4:   100577
073360  177776                          177776
073362  177777                          177777
073364  177776                          177776
073366  100577                  HXP5:   100577
073370  177776                          177776
073372  000000                          0
073374  000000                          0
073376  000252                  HXP6:   252
073400  125252                          125252
073402  125252                          125252
073404  125252                          125252

073406  000577                  HXP7:   577
073410  177776                          177776
073412  000000                          0
073414  000000                          0
073416  000577                  HXP8:   577
073420  177777                          177777
073422  000000                          0
073424  000000                          0

073426                          HXDONE:
073426  004767  031140                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
```

```
                                    ;*******************************************************************************
                                    ;TEST 473        CMPD TEST
                                    ;*******************************************************************************
073432                              TS473:

                                    ;TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)
073432  004737  074126              AAA1:   JSR     PC,@#CMPSUB
073436  000000  000000  000000      1$:     .WORD   0,0,0,0                 ;AC0
073444  000000
073446  000000  000000  000000      2$:     .WORD   0,0,0,0                 ;FSRC
073454  000000
073456  000200                      3$:     200                             ;FPS BEFORE EXECUTION
073460  000204                              204                             ;FPS AFTER EXECUTION


                                    ;TEST CMPD WITH (AC=0) AND FSRC POSITIVE.
073462  004737  074126              AAA2:   JSR     PC,@#CMPSUB
073466  000000  000000  000000      1$:     .WORD   0,0,0,0                 ;AC
073474  000000
073476  025252                      2$:     25252                           ;FSRC
073500  052525                              52525
073502  125252                              125252
073504  052525                              52525
073506  000200                      3$:     200                             ;FPS BEFORE EXECUTION
073510  000200                              200                             ;FPS AFTER EXECUTION

                                    ;TEST CMPD WITH (AC=0) AND FSRC NEGATIVE
073512  004737  074126              AAA3:   JSR     PC,@#CMPSUB
073516  000000  000000  000000      1$:     .WORD   0,0,0,0                 ;AC
073524  000000
073526  125252                      2$:     125252                          ;FSRC
073530  125252                              125252
073532  052525                              52525
073534  125252                              125252
073536  000200                      3$:     200                             ;FPS BEFORE EXECUTION
073540  000210                              210                             ;FPS AFTER EXECUTION

                                    ;TEST CMPD WITH (FSRC=0) AND AC POSITIVE
073542  004737  074126              AAA4:   JSR     PC,@#CMPSUB
073546  025252                      1$:     25252                           ;AC
073550  052525                              52525
073552  125252                              125252
073554  052525                              52525
073556  000000  000000  000000      2$:     .WORD   0,0,0,0                 ;FSRC
073564  000000
073566  000200                      3$:     200                             ;FPS BEFORE EXECUTION
073570  000210                              210                             ;FPS AFTER EXECUTION


                                    ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
073572  004737  074126              AAA5:   JSR     PC,@#CMPSUB
073576  125252                      1$:     125252                          ;AC
```

```
073600  125252                          125252
073602  052525                          52525
073604  125252                          125252
073606  000000  000000  000000  2$:     .WORD   0,0,0,0                 ;FSRC
073614  000000
073616  000200                  3$:     200                             ;FPS BEFORE EXECUTION
073620  000200                          200                             ;FPS AFTER EXECUTION


                                ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
073622  004737  074126          AAA6:   JSR     PC,@#CMPSUB
073626  052525                  1$:     52525                           ;AC
073630  125252                          125252
073632  052525                          52525
073634  125252                          125252
073636  125252                  2$:     125252                          ;;FSRC
073640  052525                          52525
073642  125252                          125252
073644  052525                          52525
073646  000200                  3$:     200                             ;FPS BEFORE EXECUTION
073650  000210                          210                             ;FPS AFTER EXECUTION



                                ;TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
073652  004737  074126          AAA7:   JSR     PC,@#CMPSUB
073656  125252                  1$:     125252                          ;AC
073660  052525                          52525
073662  125252                          125252
073664  052525                          52525
073666  052525                  2$:     52525                           ;FSRC
073670  125252                          125252
073672  052525                          52525
073674  125252                          125252
073676  000200                  3$:     200                             ;FPS BEFORE EXECUTION
073700  000200                          200                             ;FPS AFTER EXECUTION


                                ;TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
                                ;AND EAC LESS THAN EFSRC.
073702  004737  074126          AAA8:   JSR     PC,@#CMPSUB
073706  012345                  1$:     12345                           ;AC
073710  067654                          67654
073712  032101                          32101
073714  023456                          23456
073716  023456                  2$:     23456                           ;FSRC
073720  076543                          76543
073722  021012                          21012
073724  034567                          34567
073726  000200                  3$:     200                             ;FPS BEFORE EXECUTION
073730  000200                          200                             ;FPS AFTER EXECUTION



                                ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC
073732  004737  074126          AAA9:   JSR     PC,@#CMPSUB
073736  045676                  1$:     45676                           ;AC
073740  054321                          54321
073742  012345                          12345
073744  067654                          67654
```

```
073746  034567          2$:     34567                           ;FSRC
073750  065432                  65432
073752  101234                  101234
073754  056765                  56765
073756  000200          3$:     200                             ;FPS BEFORE EXECUTION
073760  000210                  210                             ;FPS AFTER EXECUTION

                        ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
073762  004737  074126  AAA10:  JSR     PC,@#CMPSUB
073766  012345          1$:     12345                           ;AC
073770  067012                  67012
073772  034567                  34567
073774  012345                  012345
073776  012345          2$:     12345                           ;FSRC
074000  067012                  67012
074002  034567                  34567
074004  012345                  012345
074006  000200          3$:     200                             ;FPS BEFORE EXECUTION
074010  000204                  204                             ;FPS AFTER EXECUTION

                        ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
                        ;AND FSRC GREATER THAN AC.
074012  004737  074126  AAA11:  JSR     PC,@#CMPSUB
074016  012345          1$:     12345                           ;AC
074020  067012                  67012
074022  034567                  34567
074024  012345                  012345
074026  012345          2$:     12345                           ;FSRC
074030  070123                  70123
074032  045670                  45670
074034  123456                  123456
074036  000200          3$:     200                             ;FPS BEFORE EXECUTION
074040  000200                  200                             ;FPS AFTER EXECUTION

                        ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
                        ;AND AC GREATER THAN FSRC.
074042  004737  074126  AAA12:  JSR     PC,@#CMPSUB
074046  054321          1$:     54321                           ;AC
074050  076543                  76543
074052  021076                  21076
074054  054321                  54321
074056  054321          2$:     54321                           ;FSRC
074060  065432                  65432
074062  107654                  107654
074064  032107                  32107
074066  000200          3$:     200                             ;FPS BEFORE EXECUTION
074070  000210                  210                             ;FPS AFTER EXECUTION

                        ;TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC,
                        ;AND AC GREATER THAN FSRC
074072  004737  074126  AAA13:  JSR     PC,@#CMPSUB
074076  112345          1$:     112345                          ;AC
074100  043210                  43210
074102  076543                  76543
074104  021076                  21076
074106  112345          2$:     112345                          ;FSRC
```

```
074110  054321                          54321
074112  007654                          07654
074114  032107                          32107
074116  000200                  3$:     200                             ;FPS BEFORE EXECUTION
074120  000210                          210                             ;FPS AFTER EXECUTION


074122  000137  074232                  JMP     @#AAADONE               ;FINISHED CMPD TEST.


                                ;THIS SUBROUTINE, CMPSUB, IS CALLED TO SET UP, EXECUTE
                                ;AND CHECK THE RESULTS OF A CMPD INSTRUCTION.
                                ;IT IS CALLED THUS:
                                ;
                                ;               JSR     PC,@#CMPSUB
                                ;       ACARG:  .WORD   X,X,X,X                 ;AC OPERAND
                                ;       FSRCARG:.WORD   X,X,X,X                 ;FSRC OPERAND
                                ;       FPSB:   .WORD   X                       ;FPS BEFORE EXECUTION
                                ;       FPSA:   .WORD   X                       ;FPS AFTER EXECUTION
                                ;       FPSE:   .WORD   X                       ;ERROR FPS
                                ;       ERR:    ERROR   X                       ;FPS ERROR
                                ;       CONT:                                   ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
                                ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, CMPD, IS EXECUTED.
                                ;AFTER THE EXECUTION THE FPS IS CHECKED AGAINST FPSA. IF IT IS A MATCH
                                ;THEN THERE WAS NO ERROR AND CONTROL IS RETURNED TO CONT. IF
                                ;THE FPS IS INCORRECT IT IS COMPARED WITH FPSE IN AN ATTEMPT TO ANALYSE
                                ;THE FAILURE. IF THE FPS IS THE SAME AS FPSE THEN CONTROL IS
                                ;RETURNED TO THE ERROR CALL AT LOCATION ERR. IF THE FPS WAS
                                ;NOT CORRECT BUT DIDN'T MATCH FPSE A GENERAL ERROR IS REPORTED
                                ;AND CONTROL IS PASSED TO CONT.

074126  012601                  CMPSUB: MOV     (SP)+,R1                ;PICK UP A POINTER TO THE
                                                                        ;ARGUMENTS.
074130  016100  000020                  MOV     20(R1),R0               ;GET THE FPS BEFORE EXECUTION.
074134  170100                          LDFPS   R0                      ;LOAD IT INTO THE FPS.

074136  010100                          MOV     R1,R0                   ;GET ADDRESS OF AC OPERAND.
074140  172410                          LDD     (R0),AC0                ;LOAD AC0 OPERAND

074142  010100                          MOV     R1,R0                   ;COMPUTE FSRC OPERAND
074144  062700  000010                  ADD     #10,R0                  ;ADDRESS

074150  000240                          NOP                             ;FOR SCOPING.
074152  173410                  1$:     CMPD    (R0),AC0                ;EXECUTE THE TEST INSTRUCTION.

074154  170205                          STFPS   R5                      ;SAVE FPS AFTER INSTRUCTION.

074156  016104  000022                  MOV     22(R1),R4               ;GET EXPECTED FPS.
074162  020405                          CMP     R4,R5                   ;WAS FPS CORRECT?
074164  020405                          CMP     R4,R5                   ;WAS FPS CORRECT?
074166  001401                          BEQ     3$
074170  104000                          EMT                     ;
074172  012700  074222          3$:     MOV     #CMPTMP,R0              ;IF FPS WAS CORRECT MAKE SURE
074176  174010                          STD     AC0,(R0)                ;AC0 WAS NOT AFFECTED BY CMPD.
```

```
074200  010102                          MOV     R1,R2
074202  012703  000004                  MOV     #4,R3
074206  022220                  4$:     CMP     (R2)+,(R0)+
074210  001401                          BEQ     5$
074212  104000                          EMT                             ;
074214  077304                  5$:     SOB     R3,4$
074216  000161  000024                  JMP     24(R1)                          ;RETURN
074222  000000  000000  000000  CMPTMP: .WORD   0,0,0,0
074230  000000
074232                          AAADONE:
074232  004767  030334                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 474       DIVD WITH (FSRC=0) AND (BUT FD) TEST
                                ;*******************************************************************************
074236                          TS474:
074236  012704  040200          BBB0:   MOV     #40200,R4               ;SET UP FPS
                                                                        ;WITH INTERRUPTS
                                                                        ;DISABLED.
074242  170104                          LDFPS   R4
074244  012737  074302  000244          MOV     #BBBER1,@#FPVECT;SET UP FOR ANY FP INTERRUPTS.
074252  012700  074466                  MOV     #BBBP1,R0               ;SET UP AC0 = 0
074256  172410                          LDD     (R0),AC0
074260  012701  074466                  MOV     #BBBP1,R1               ;FSRC = 0
074264  174411                  BBB1:   DIVD    (R1),AC0                ;TEST INSTRUCTION
074266  170205                          STFPS   R5                      ;GET FPS
074270  170303                          STST    R3                      ;GET FEC
074272  012704  140204                  MOV     #140204,R4              ;EXPECTED FPS.
074276  020405                          CMP     R4,R5                   ;IS FPS CORRECT.
074300  001401                          BEQ     BBB7
074302                          BBBER1:
074302  104000                          EMT                             ;
074304  012702  000004          BBB7:   MOV     #4,R2                   ;EXPECTED FEC.
074310  020203                          CMP     R2,R3                   ;IS FEC CORRECT?
074312  001401                          BEQ     BBB2
074314  104000                          EMT                             ;
                                ;TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
074316  012704  040200          BBB2:   MOV     #40200,R4               ;LOAD FPS WITH TRAPS DISABLED.
074322  170104                          LDFPS   R4
```

```

074324 012700 074476                MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO).
074330 172410                       LDD     (R0),AC0
074332 012700 074466                MOV     #BBBP1,R0       ;FSRC=0
074336 174410               BBB3:   DIVD    (R0),AC0

074340 170205                       STFPS   R5              ;GET FPS.
074342 170303                       STST    R3              ;GET FEC.

074344 012704 140200                MOV     #140200,R4              ;EXPECTED FPS.
074350 020405                       CMP     R4,R5           ;IS FPS CORRECT?
074352 001401                       BEQ     1$
074354 104000                       EMT                     ;
074356 012702 000004        1$:     MOV     #4,R2           ;EXPECTED FEC.
074362 020203                       CMP     R2,R3           ;WAS FEC CORRECT?
074364 001401                       BEQ     BBB4
074366 104000                       EMT                     ;

                            ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
074370 012704 000200        BBB4:   MOV     #200,R4         ;SET UP FPS. TRAP ENABLED.
074374 170104                       LDFPS   R4

074376 012700 074476                MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO).
074402 172410                       LDD     (R0),AC0

074404 012737 074424 000244         MOV     #BBB6,@#FPVECT  ;SET UP FOR THE EXPECTED INTERRUPT.
074412 012700 074466                MOV     #BBBP1,R0       ;FSRC=0

074416 174410               BBB5:   DIVD    (R0),AC0        ;TEST INSTRUCTION (SHOULD RESULT IN TRAP).
074420 170000                       CFCC
074422 104000                       EMT                     ;
074424 022716 074420        BBB6:   CMP     #BBB5+2,(SP)    ;TRAP TO HERE WHEN THE DIVISION BY 0
                                                            ;OCCURS. FIRST SEE IF THE ADDRESS OF
                                                            ;THE TRAP IS 2+THE ADDRESS OF THE TEST
                                                            ;DIVD INSTRUCTION.
074430 001401                       BEQ     1$
074432 104000                       EMT                     ;
074434 170205               1$:     STFPS   R5              ;GET FPS.
074436 170303                       STST    R3              ;GET FEC.
074440 022626                       CMP     (SP)+,(SP)+     ;RESET THE STACK.

074442 012704 100200                MOV     #100200,R4      ;EXPECTED FPS.
074446 020405                       CMP     R4,R5           ;IS FPS CORRECT?
074450 001401                       BEQ     2$
074452 104000                       EMT                     ;
074454 012702 000004        2$:     MOV     #4,R2           ;EXPECTED FEC.
074460 020203                       CMP     R2,R3           ;IS FEC CORRECT?
074462 001411                       BEQ     BBBDONE
074464 104000                       EMT                     ;

074466 000000 000000 000000 BBBP1:  .WORD   0,0,0,0
074474 000000
074476 012345 054321 023456 BBBP2:  .WORD   12345,54321,23456,76543
074504 076543


```

```
074506                          BBBDONE:
074506  004767  030060                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).



                                ;*******************************************************************************
                                ;TEST 475       DIVF TEST
                                ;*******************************************************************************
074512                          TS475:

                                ;CHECK DIVF WITH (AC=0).
074512  004767  000404          CCC1:   JSR     PC,DIVFSUB
074516  000000  000000          1$:     .WORD   0,0                     ;AC
074522  012345  067012          2$:     .WORD   12345,67012             ;FSRC
074526  000000  000000          3$:     .WORD   0,0                     ;RES
074532  000000                  4$:     0                               ;FPS BEFORE EXECUTION
074534  000004                          4                               ;FPS AFTER EXECUTION

                                ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.
074536  004737  075122          CCC2:   JSR     PC,@#DIVFSUB
074542  065652  125252          1$:     .WORD   65652,125252            ;AC
074546  065600  000000          2$:     .WORD   65600,0                 ;FSRC
074552  040252  125252          3$:     .WORD   40252,125252            ;RES
074556  003000                  4$:     3000                            ;FPS BEFORE EXECUTION.
074560  003000                          3000                            ;FPS AFTER EXECUTION.

                                ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.
074562  004767  000334          CCC3:   JSR     PC,DIVFSUB
074566  076400  000000          1$:     .WORD   76400,0                 ;AC
074572  076400  000000          2$:     .WORD   76400,0                 ;FSRC
074576  040200  000000          3$:     .WORD   40200,0                 ;RES
074602  001000                  4$:     1000                            ;FPS BEFORE EXECUTION.
074604  001000                          1000                            ;FPS AFTER EXECUTION.
                                ;TEST DIVF WITH BOTH OPERANDS POSITIVE.
074606  004737  075122          CCC4:   JSR     PC,@#DIVFSUB
074612  056777  177777          1$:     .WORD   56777,177777            ;AC
074616  054200  000000          2$:     .WORD   54200,0                 ;FSRC
074622  042777  177777          3$:     .WORD   42777,177777            ;RES
074626  000000                  4$:     0                               ;FPS BEFORE EXECUTION.
074630  000000                          0                               ;FPS AFTER EXECUTION.

                                ;TEST THE DIVF INSTRUCTION:
074632  004737  075122          CCC5:   JSR     PC,@#DIVFSUB
074636  012377  177777          1$:     .WORD   12377,177777            ;AC
074642  012300  000000          2$:     .WORD   12300,0                 ;FSRC
074646  040252  125252          3$:     .WORD   40252,125252            ;RES
074652  000000                  4$:     0                               ;FPS BEFORE EXECUTION.
074654  000000                          0                               ;FPS AFTER EXECUTION.

                                ;TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
074656  004737  075122          CCC6:   JSR     PC,@#DIVFSUB
```

```
16425  074662  064600  000001        1$:     .WORD   64600,1          ;AC
```

```
074666  066600  000000          2$:     .WORD   66600,0         ;FSRC
074672  036200  000001          3$:     .WORD   36200,1         ;RES
074676  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
074700  000000                          0                       ;FPS AFTER EXECUTION.

                                ;TEST DIVF.
074702  004737  075122          CCC7:   JSR     PC,@#DIVFSUB
074706  034577  177776          1$:     .WORD   34577,177776    ;AC
074712  023400  000000          2$:     .WORD   23400,0         ;FSRC
074716  051377  177776          3$:     .WORD   51377,177776    ;RES
074722  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
074724  000000                          0                       ;FPS AFTER EXECUTION.


                                ;DIVF TEST.
074726  004737  075122          CCC8:   JSR     PC,@#DIVFSUB
074732  067652  125252          1$:     .WORD   67652,125252    ;AC
074736  056500  000000          2$:     .WORD   56500,0         ;FSRC
074742  051343  107070          3$:     .WORD   51343,107070    ;RES
074746  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
074750  000000                          0                       ;FPS AFTER EXECUTION.

                                ;DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
074752  004737  075122          CCC9:   JSR     PC,@#DIVFSUB
074756  140400  000000          1$:     .WORD   140400,0        ;AC
074762  140500  000000          2$:     .WORD   140500,0        ;FSRC
074766  040052  125253          3$:     .WORD   040052,125253   ;RES
074772  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
074774  000000                          0                       ;FPS AFTER EXECUTION.

                                ;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
074776  004737  075122          CCC10:  JSR     PC,@#DIVFSUB
075002  160077  000000          1$:     .WORD   160077,0        ;AC
075006  040277  000000          2$:     .WORD   40277,0         ;FSRC
075012  160000  000000          3$:     .WORD   160000,0        ;RES
075016  000007                  4$:     7                       ;FPS BEFORE EXECUTION.
075020  000010                          10                      ;FPS AFTER EXECUTION.

                                ;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
075022  004737  075122          CCC11:  JSR     PC,@#DIVFSUB
075026  040400  000000          1$:     .WORD   40400,0         ;AC
075032  140500  000000          2$:     .WORD   140500,0        ;FSRC
075036  140052  125253          3$:     .WORD   140052,125253   ;RES
075042  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
075044  000010                          10                      ;FPS AFTER EXECUTION.


                                ;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
075046  004737  075122          CCC12:  JSR     PC,@#DIVFSUB
075052  060100  000001          1$:     .WORD   60100,1         ;AC
075056  040300  000000          2$:     .WORD   40300,0         ;FSRC
075062  060000  000000          3$:     .WORD   60000,0         ;RES
075066  000052                  4$:     52                      ;FPS BEFORE EXECUTION.
075070  000040                          40                      ;FPS AFTER EXECUTION.

                                ;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.
```

```
075072  004767  000024      CCC13:  JSR     PC,DIVFSUB
075076  060100  000001      1$:     .WORD   60100,1         ;AC
075102  040300  000000      2$:     .WORD   40300,0         ;FSRC
075106  060000  000001      3$:     .WORD   60000,1         ;RES
075112  000005              4$:     5                       ;FPS BEFORE EXECUTION.
075114  000000                      0                       ;FPS AFTER EXECUTION.

075116  000137  075240              JMP     @#CCCDONE       ;GO TO NEXT TEST.

                            ;THIS SUBROUTINE, DIVFSUB, IS CALLED TO SET UP, EXECUTE
                            ;AND CHECK THE RESULT OF A DIVF INSTRUCTION. IT IS CALLED THUS:
                            ;
                            ;                JSR      PC,@#DIVFSUB
                            ;       ACARG:   .WORD    X,X                        ;AC OPERAND
                            ;       FSRCARG:.WORD     X,X                        ;FSRC OPERAND
                            ;       RES:     .WORD    X,X                        ;EXPECTED RESULT
                            ;       FPSB:    .WORD    X                          ;FPS BEFORE EXECUTION
                            ;       FPSA:    .WORD    X                          ;FPS AFTER EXECUTION
                            ;       ERRES:   .WORD    X,X                        ;ERROR RESULT
                            ;       ERR:     ERROR    X                          ;RESULT ERROR
                            ;       CONT:                                        ;RETURN ADDRESS
                            ;
                            ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
                            ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVF IS EXECUTED.
                            ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
                            ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
                            ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
                            ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
                            ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
                            ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
                            ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
                            ;THEN THE FAILURE IS REPORTED IN DIVFSUB AND CONTROL IS PASSED TO
                            ;CONT. IF NO ERRORS ARE DETECTED THEN DIVFSUB RETURNS CONTROL
                            ;TO CONT.

075122  012601              DIVFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
075124  012700  000200              MOV     #200,R0         ;SET FD MODE.
075130  170100                      LDFPS   R0
075132  010100                      MOV     R1,R0           ;LOAD THE AC OPERAND.
075134  172410                      LDD     (R0),AC0
075136  016100  000014              MOV     14(R1),R0       ;LOAD THE FPS
075142  170100                      LDFPS   R0
075144  010100                      MOV     R1,R0
075146  062700  000004              ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.

075152  174410              1$:     DIVF    (R0),AC0        ;TEST INSTRUCTION.

075154  170204                      STFPS   R4              ;GET THE FPS.
075156  012700  000200              MOV     #200,R0         ;SET FD MODE
075162  170100                      LDFPS   R0

075164  012700  075230              MOV     #DIVFT,R0       ;GET THE RESULT OF THE DIVF.
075170  174010                      STD     AC0,(R0)
075172  021061  000010              CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
075176  001401                      BEQ     2$
075200  104000                      EMT                     ;
```

```
075202  026061  000002  000012  2$:     CMP     2(R0),12(R1)
075210  001401                          BEQ     3$
075212  104000                          EMT                     ;
075214  026104  000016          3$:     CMP     16(R1),R4       ;IS FPS CORRECT?
075220  001401                          BEQ     4$
075222  104000                          EMT                     ;
075224  000161  000020          4$:     JMP     20(R1)          ;IF NO ERRORS OCCURRED RETURN.

075230  000000  000000  000000  DIVFT:  .WORD   0,0,0,0
075236  000000

075240                          CCCDONE:
075240  004767  027326                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 476       DIVD TEST
                                ;*******************************************************************************
075244                          TS476:

                                ;DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.
075244  004737  075610          DDD1:   JSR     PC,@#DIVDSUB
075250  034277  000000  000000  1$:     .WORD   34277,0,0,0     ;AC
075256  000000
075260  040277  000000  000000  2$:     .WORD   40277,0,0,0     ;FSRC
075266  000000
075270  034200  000000  000000  3$:     .WORD   34200,0,0,0     ;RES
075276  000000
075300  000200                  4$:     200                     ;FPS BEFORE EXECUTION.
075302  000200                          200                     ;FPS AFTER EXECUTION.

                                ;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.
075304  004737  075610          DDD2:   JSR     PC,@#DIVDSUB
075310  134277  000000  000000  1$:     .WORD   134277,0,0,0    ;AC
075316  000000
075320  040277  000000  000000  2$:     .WORD   40277,0,0,0     ;FSRC
075326  000000
075330  134200  000000  000000  3$:     .WORD   134200,0,0,0    ;RES
075336  000000
075340  000207                  4$:     207                     ;FPS BEFORE EXECUTION.
075342  000210                          210                     ;FPS AFTER EXECUTION.

                                ;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
075344  004767  000240          DDD3:   JSR     PC,DIVDSUB
075350  134300  000000  000000  1$:     .WORD   134300,0,0,1    ;AC
075356  000001
075360  140300  000000  000000  2$:     .WORD   140300,0,0,0    ;FSRC
075366  000000
075370  034200  000000  000000  3$:     .WORD   34200,0,0,0     ;RES
075376  000000
075400  000250                  4$:     250                     ;FPS BEFORE EXECUTION.
```

```
075402  000240                                 240                                 ;FPS AFTER EXECUTION.

                                       ;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
075404  004737  075610                 DDD4:   JSR     PC,@#DIVDSUB
075410  034300  000000  000000         1$:     .WORD   34300,0,0,1         ;AC
075416  000001
075420  140300  000000  000000         2$:     .WORD   140300,0,0,0        ;FSRC
075426  000000
075430  134200  000000  000000         3$:     .WORD   134200,0,0,1        ;RES
075436  000001
075440  000207                         4$:     207                         ;FPS BEFORE EXECUTION.
075442  000210                                 210                         ;FPS AFTER EXECUTION.

                                       ;DIVD TEST.
075444  004737  075610                 DDD5:   JSR     PC,@#DIVDSUB
075450  100400  000000  000000         1$:     .WORD   100400,0,0,0        ;AC
075456  000000
075460  000500  000000  000000         2$:     .WORD   500,0,0,0           ;FSRC
075466  000000
075470  140052  125252                 3$:     .WORD   140052,125252       ;RES
075474  125252  125252                         .WORD   125252,125252
075500  007647                         4$:     7647                        ;FPS BEFORE EXECUTION.
075502  007650                                 7650                        ;FPS AFTER EXECUTION.


                                       ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
075504  004737  075610                 DDD6:   JSR     PC,@#DIVDSUB
075510  000400  000000  000000         1$:     .WORD   400,0,0,0           ;AC
075516  000000
075520  100500  000000  000000         2$:     .WORD   100500,0,0,0        ;FSRC
075526  000000
075530  140052  125252                 3$:     .WORD   140052,125252       ;RES
075534  125252  125253                         .WORD   125252,125253
075540  007707                         4$:     7707                        ;FPS BEFORE EXECUTION.
075542  007710                                 7710                        ;FPS AFTER EXECUTION.

                                       ;DIVD TEST.
075544  004737  075610                 DDD7:   JSR     PC,@#DIVDSUB
075550  170360  170360                 1$:     .WORD   170360,170360       ;AC
075554  170360  170360                         .WORD   170360,170360
075560  170360  170360                 2$:     .WORD   170360,170360       ;FSRC
075564  170360  170360                         .WORD   170360,170360
075570  040200  000000  000000         3$:     .WORD   40200,0,0,0         ;RES
075576  000000
075600  007717                         4$:     7717                        ;FPS BEFORE EXECUTION.
075602  007700                                 7700                        ;FPS AFTER EXECUTION.

075604  000137  075732                         JMP     @#DDDDONE           ;GO TO NEXT TEST.



                                       ;THIS SUBROUTINE, DIVDSUB, IS CALLED TO SET UP, EXECUTE
                                       ;AND CHECK THE RESULT OF A DIVD INSTRUCTION. IT IS CALLED THUS:
                                       ;
                                       ;                JSR     PC,@#DIVDSUB
                                       ;       ACARG:   .WORD   X,X,X,X                 ;AC OPERAND
                                       ;       FSRCARG:.WORD    X,X,X,X                 ;FSRC OPERAND
```

```
                                   ;        RES:     .WORD    X,X,X,X              ;EXPECTED RESULT
                                   ;        FPSB:    .WORD    X                    ;FPS BEFORE EXECUTION
                                   ;        FPSA:    .WORD    X                    ;FPS AFTER EXECUTION
                                   ;        ERRES:   .WORD    X,X,X,X              ;ERROR RESULT
                                   ;        ERR:     ERROR    X                    ;RESULT ERROR
                                   ;        CONT:                                  ;RETURN ADDRESS
                                   ;
                                   ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
                                   ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVD IS EXECUTED.
                                   ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
                                   ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
                                   ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
                                   ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
                                   ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
                                   ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
                                   ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
                                   ;THEN THE FAILURE IS REPORTED IN DIVDSUB AND CONTROL IS PASSED TO
                                   ;CONT. IF NO ERRORS ARE DETECTED THEN DIVDSUB RETURNS CONTROL
                                   ;TO CONT.

075610  012601                     DIVDSUB:          MOV      (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
075612  012700  000200                      MOV      #200,R0          ;SET FD MODE.
075616  170100                              LDFPS    R0

075620  010100                              MOV      R1,R0            ;SET UP THE AC0 OPERAND.
075622  172410                              LDD      (R0),AC0
075624  016100  000030                      MOV      30(R1),R0        ;LOAD THE FPS.
075630  170100                              LDFPS    R0

075632  010100                              MOV      R1,R0            ;ESTABLISH A POINTER TO FSRC.
075634  062700  000010                      ADD      #10,R0

075640  174410                     1$:      DIVD     (R0),AC0         ;EXECUTE THE TEST INSTRUCTION.
075642  170204                              STFPS    R4               ;GET THE FPS.
075644  012700  000200                      MOV      #200,R0          ;SET FD MODE.
075650  170100                              LDFPS    R0
075652  012700  075722                      MOV      #DIVDT,R0        ;GET THE RESULT.
075656  174010                              STD      AC0,(R0)
075660  010102                              MOV      R1,R2            ;CHECK THE RESULT.
075662  062702  000020                      ADD      #20,R2
075666  012703  075722                      MOV      #DIVDT,R3
075672  012705  000004                      MOV      #4,R5
075676  022223                     2$:      CMP      (R2)+,(R3)+
075700  001401                              BEQ      3$
075702  104000                              EMT                       ;
075704  077504                     3$:      SOB      R5,2$

075706  026104  000032                      CMP      32(R1),R4        ;IS FPS CORRECT?
075712  001401                              BEQ      4$
075714  104000                              EMT                       ;
075716  000161  000034             4$:      JMP      34(R1)           ;RETURN.
075722  000000  000000  000000     DIVDT:   .WORD    0,0,0,0
075730  000000

075732                             DDDDONE:
075732  004767  026634                      JSR      PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
```

```
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                        ;THE USER TYPED CONTROL G?).



                        ;*******************************************************************
                        ;TEST 477       MULF TEST
                        ;*******************************************************************
075736                  TS477:

                        ;MULF WITH (FSRC=AC=0)
075736 004737 076346    EEE1:   JSR     PC,@#MULFSUB
075742 000000 000000    1$:     .WORD   0,0             ;AC
075746 000000 000000    2$:     .WORD   0,0             ;FSRC
075752 000000 000000    3$:     .WORD   0,0             ;RES
075756 007517           4$:     7517                    ;FPS BEFORE EXECUTION.
075760 007504                   7504                    ;FPS AFTER EXECUTION.

                        ;MULF WITH (FSRC=0).
075762 004737 076346    EEE2:   JSR     PC,@#MULFSUB
075766 071625 034435    1$:     .WORD   71625,34435     ;AC
075772 000000 000000    2$:     .WORD   0,0             ;FSRC
075776 000000 000000    3$:     .WORD   0,0             ;RES
076002 000013           4$:     13                      ;FPS BEFORE EXECUTION.
076004 000004                   4                       ;FPS AFTER EXECUTION.

                        ;MULF WITH (AC=0)
076006 004737 076346    EEE3:   JSR     PC,@#MULFSUB
076012 000000 000000    1$:     .WORD   0,0             ;AC
076016 071625 153443    2$:     .WORD   071625,153443   ;FSRC
076022 000000 000000    3$:     .WORD   0,0             ;RES
076026 007500           4$:     7500                    ;FPS BEFORE EXECUTION.
076030 007504                   7504                    ;FPS AFTER EXECUTION.

                        ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
076032 004737 076346    EEE4:   JSR     PC,@#MULFSUB
076036 040200 000000    1$:     .WORD   40200,0         ;AC
076042 040177 177777    2$:     .WORD   40177,-1        ;FSRC
076046 040177 177777    3$:     .WORD   40177,-1        ;RES
076052 000017           4$:     17                      ;FPS BEFORE EXECUTION.
076054 000000                   0                       ;FPS AFTER EXECUTION.

                        ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.
076056 004767 000264    EEE5:   JSR     PC,MULFSUB
076062 040177 177777    1$:     .WORD   40177,-1        ;AC
076066 040200 000000    2$:     .WORD   40200,0         ;FSRC
076072 040177 177777    3$:     .WORD   40177,-1        ;RES
076076 000040           4$:     40                      ;FPS BEFORE EXECUTION.
076100 000040                   40                      ;FPS AFTER EXECUTION.

                        ;MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
076102 004737 076346    EEE6:   JSR     PC,@#MULFSUB
076106 040100 000000    1$:     .WORD   40100,0         ;AC
076112 040100 000000    2$:     .WORD   40100,0         ;FSRC
```

```
076116  040020  000000    3$:     .WORD   40020,0         ;RES
076122  000012            4$:     12                      ;FPS BEFORE EXECUTION.
076124  000000                    0                       ;FPS AFTER EXECUTION.

                          ;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
076126  004737  076346    EEE7:   JSR     PC,@#MULFSUB
076132  017500  000000    1$:     .WORD   17500,0         ;AC
076136  023652  125252    2$:     .WORD   23652,125252    ;FSRC
076142  003177  177777    3$:     .WORD   3177,-1         ;RES
076146  007417            4$:     7417                    ;FPS BEFORE EXECUTION.
076150  007400                    7400                    ;FPS AFTER EXECUTION.

                          ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
076152  004737  076346    EEE8:   JSR     PC,@#MULFSUB
076156  040342  000000    1$:     .WORD   40342,0         ;AC
076162  176542  000000    2$:     .WORD   176542,0        ;FSRC
076166  176707  102000    3$:     .WORD   176707,102000   ;RES
076172  000007            4$:     7                       ;FPS BEFORE EXECUTION.
076174  000010                    10                      ;FPS AFTER EXECUTION.

                          ;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
076176  004737  076346    EEE9:   JSR     PC,@#MULFSUB
076202  140200  000000    1$:     .WORD   140200,0        ;AC
076206  007417  007417    2$:     .WORD   7417,7417       ;FSRC
076212  107417  007417    3$:     .WORD   107417,7417     ;RES
076216  000000            4$:     0                       ;FPS BEFORE EXECUTION.
076220  000010                    10                      ;FPS AFTER EXECUTION.

                          ;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
076222  004737  076346    EEE10:  JSR     PC,@#MULFSUB
076226  144600  000000    1$:     .WORD   144600,0        ;AC
076232  154000  000000    2$:     .WORD   154000,0        ;FSRC
076236  060400  000000    3$:     .WORD   60400,0         ;RES
076242  000017            4$:     17                      ;FPS BEFORE EXECUTION.
076244  000000                    0                       ;FPS AFTER EXECUTION.

                          ;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
076246  004737  076346    EEE11:  JSR     PC,@#MULFSUB
076252  140300  000000    1$:     .WORD   140300,0        ;AC
076256  160000  000001    2$:     .WORD   160000,1        ;FSRC
076262  060100  000002    3$:     .WORD   60100,2         ;RES
076266  000010            4$:     10                      ;FPS BEFORE EXECUTION.
076270  000000                    0                       ;FPS AFTER EXECUTION.

                          ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
076272  004737  076346    EEE12:  JSR     PC,@#MULFSUB
076276  060000  000001    1$:     .WORD   60000,1         ;AC
076302  140300  000000    2$:     .WORD   140300,0        ;FSRC
076306  160100  000001    3$:     .WORD   160100,1        ;RES
076312  007547            4$:     7547                    ;FPS BEFORE EXECUTION.
076314  007550                    7550                    ;FPS AFTER EXECUTION.

                          ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
076316  004737  076346    EEE13:  JSR     PC,@#MULFSUB
076322  040277  000000    1$:     .WORD   40277,0         ;AC
076326  060000  000001    2$:     .WORD   60000,1         ;FSRC
```

```
076332  060077  000001          3$:     .WORD   60077,1         ;RES
076336  000014                  4$:     14                      ;FPS BEFORE EXECUTION.
076340  000000                          0                       ;FPS AFTER EXECUTION.

076342  000167  000116                  JMP     EEEDONE         ;GO TO THE NEXT TEST.

                                ;THIS SUBROUTINE, MULFSUB, IS CALLED TO SET UP, EXECUTE
                                ;AND CHECK THE RESULT OF A MULF INSTRUCTION. IT IS CALLED THUS:
                                ;
                                ;               JSR     PC,@#MULFSUB
                                ;       ACARG:  .WORD   X,X                     ;AC OPERAND
                                ;       FSRCARG:.WORD   X,X                     ;FSRC OPERAND
                                ;       RES:    .WORD   X,X                     ;EXPECTED RESULT
                                ;       FPSB:   .WORD   X                       ;FPS BEFORE EXECUTION
                                ;       FPSA:   .WORD   X                       ;FPS AFTER EXECUTION
                                ;       ERRES:  .WORD   X,X                     ;ERROR RESULT
                                ;       ERR:    ERROR   X                       ;RESULT ERROR
                                ;       CONT:                                   ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
                                ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULF IS EXECUTED.
                                ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
                                ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
                                ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
                                ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
                                ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
                                ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
                                ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
                                ;THEN THE FAILURE IS REPORTED IN MULFSUB AND CONTROL IS PASSED TO
                                ;CONT. IF NO ERRORS ARE DETECTED THEN MULFSUB RETURNS CONTROL
                                ;TO CONT.

076346  012601                  MULFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
076350  012700  000200                  MOV     #200,R0         ;SET FD MODE.
076354  170100                          LDFPS   R0
076356  010100                          MOV     R1,R0           ;LOAD THE AC OPERAND.
076360  172410                          LDD     (R0),AC0
076362  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
076366  170100                          LDFPS   R0
076370  010100                          MOV     R1,R0
076372  062700  000004                  ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.

076376  171010                  1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.

076400  170204                          STFPS   R4              ;GET THE FPS.
076402  012700  000200                  MOV     #200,R0         ;SET FD MODE
076406  170100                          LDFPS   R0

076410  012700  076454                  MOV     #MULFT,R0       ;GET THE RESULT OF THE MULF.
076414  174010                          STD     AC0,(R0)
076416  021061  000010                  CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
076422  001401                          BEQ     2$
076424  104000                          EMT                     ;
076426  026061  000002  000012  2$:     CMP     2(R0),12(R1)
076434  001401                          BEQ     3$
076436  104000                          EMT                     ;
```

```
076440  026104  000016          3$:     CMP     16(R1),R4       ;IS FPS CORRECT?
076444  001401                          BEQ     4$
076446  104000                          EMT                     ;
076450  000161  000020          4$:     JMP     20(R1)          ;IF NO ERRORS OCCURRED RETURN.

076454  000000  000000  000000  MULFT:  .WORD   0,0,0,0
076462  000000

076464                          EEEDONE:
076464  004767  026102                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).



                                ;*******************************************************************
                                ;TEST 500       MULD TEST
                                ;*******************************************************************
076470                          TS500:

                                ;MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.
076470  004737  076674          FFF1:   JSR     PC,@#MULDSUB
076474  040200  000000  000000  1$:     .WORD   40200,0,0,0     ;AC
076502  000000
076504  023777  177777  177777  2$:     .WORD   23777,-1,-1,-1  ;FSRC
076512  177777
076514  023777  177777  177777  3$:     .WORD   23777,-1,-1,-1  ;RES
076522  177777
076524  000217                  4$:     217                     ;FPS BEFORE EXECUTION.
076526  000200                          200                     ;FPS AFTER ECECUTION.

                                ;MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
076530  004767  000140          FFF2:   JSR     PC,MULDSUB
076534  065400  000000  000000  1$:     .WORD   65400,0,0,1     ;AC
076542  000001
076544  037577  177777  177777  2$:     .WORD   37577,-1,-1,-2  ;FSRC
076552  177776
076554  064777  177777  177777  3$:     .WORD   64777,-1,-1,-1  ;RES
076562  177777
076564  000247                  4$:     247                     ;FPS BEFORE EXECUTION.
076566  000240                          240                     ;FPS AFTER EXECUTION.

                                ;MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
076570  004737  076674          FFF3:   JSR     PC,@#MULDSUB
076574  137577  177777  177777  1$:     .WORD   137577,-1,-1,-2 ;AC
076602  177776
076604  165400  000000  000000  2$:     .WORD   165400,0,0,1    ;FSRC
076612  000001
076614  065000  000000  000000  3$:     .WORD   65000,0,0,0     ;RES
076622  000000
076624  007717                  4$:     7717                    ;FPS BEFORE EXECUTION.
076626  007700                          7700                    ;FPS AFTER EXECUTION.

                                ;MULD   TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
```

```
076630  004737  076674            FFF4:   JSR     PC,@#MULDSUB
076634  017500  000000  000000    1$:     .WORD   17500,0,0,0       ;AC
076642  000000
076644  123652  125252            2$:     .WORD   123652,125252     ;FSRC
076650  125252  125252                    .WORD   125252,125252
076654  103177  177777  177777    3$:     .WORD   103177,-1,-1,-1 ;RES
076662  177777
076664  000200                    4$:     200                       ;FPS BEFORE EXECUTION.
076666  000210                            210                       ;FPS AFTER EXECUTION.

076670  000167  000122                    JMP     FFFDONE

                                  ;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
                                  ;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:;
                                  ;               JSR     PC,@#MULDSUB
                                  ;       ACARG:  .WORD   X,X,X,X                 ;AC OPERAND
                                  ;       FSRCARG:.WORD   X,X,X,X                 ;FSRC OPERAND
                                  ;       RES:    .WORD   X,X,X,X                 ;EXPECTED RESULT
                                  ;       FPSB:   .WORD   X                       ;FPS BEFORE EXECUTION
                                  ;       FPSA:   .WORD   X                       ;FPS AFTER EXECUTION
                                  ;       ERRES:  .WORD   X,X,X,X                 ;ERROR RESULT
                                  ;       ERR:    ERROR   X                       ;RESULT ERROR
                                  ;       CONT:                                   ;RETURN ADDRESS
                                  ;
                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULD IS EXECUTED.
                                  ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
                                  ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
                                  ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
                                  ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
                                  ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
                                  ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
                                  ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
                                  ;THEN THE FAILURE IS REPORTED IN MULDSUB AND CONTROL IS PASSED TO
                                  ;CONT. IF NO ERRORS ARE DETECTED THEN MULDSUB RETURNS CONTROL
                                  ;TO CONT.

076674  012601                    MULDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
076676  012700  000200                    MOV     #200,R0           ;SET FD MODE.
076702  170100                            LDFPS   R0

076704  010100                            MOV     R1,R0             ;SET UP THE AC0 OPERAND.
076706  172410                            LDD     (R0),AC0
076710  016100  000030                    MOV     30(R1),R0         ;LOAD THE FPS.
076714  170100                            LDFPS   R0

076716  010100                            MOV     R1,R0             ;ESTABLISH A POINTER TO FSRC.
076720  062700  000010                    ADD     #10,R0

076724  171010                    1$:     MULD    (R0),AC0          ;EXECUTE THE TEST INSTRUCTION.

076726  170204                            STFPS   R4                ;GET THE FPS.
076730  012700  000200                    MOV     #200,R0           ;SET FD MODE.
076734  170100                            LDFPS   R0

076736  012700  077006                    MOV     #MULDT,R0         ;GET THE RESULT.
```

```
076742  174010                              STD     AC0,(R0)
076744  010102                              MOV     R1,R2                   ;CHECK THE RESULT.
076746  062702  000020                      ADD     #20,R2
076752  012703  077006                      MOV     #MULDT,R3
076756  012705  000004                      MOV     #4,R5
076762  022223                      2$:     CMP     (R2)+,(R3)+
076764  001401                              BEQ     3$
076766  104000                              EMT                             ;
076770  077504                      3$:     SOB     R5,2$

076772  026104  000032                      CMP     32(R1),R4               ;IS FPS CORRECT?
076776  001401                              BEQ     4$
077000  104000                              EMT                             ;
077002  000161  000034              4$:     JMP     34(R1)                  ;RETURN.

077006  000000  000000  000000      MULDT:  .WORD   0,0,0,0
077014  000000
077016                              FFFDONE:
077016  004767  025550                      JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                            ;SEE IF THE USER HAS EXPRESSED
                                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                            ;THE USER TYPED CONTROL G?).




                                    ;*************************************************************************
                                    ;TEST 501       UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST
                                    ;*************************************************************************
077022                              TS501:

                                    ;UNDERFLOW, WITH EXPONENT OF RESULT = -129
077022  004737  077166              III1:   JSR     PC,@#OVUNFNT
077026  020200  000000              1$:     .WORD   20200,0                 ;AC
077032  020000  000000              2$:     .WORD   20000,0                 ;FSRC
077036  000000  000000              3$:     .WORD   0,0                     ;RES
077042  000000                      5$:     0                               ;FPS BEFORE EXECUTION.
077044  000004                              4                               ;FPS AFTER EXECUTION.
077046  000012                      6$:     12                              ;FEC
077050  177777                              -1                              ;FLAG

                                    ;UNDERFLOW, WITH EXPONENT OF RESULT = -193
077052  004737  077166              III2:   JSR     PC,@#OVUNFNT
077056  010200  000000              1$:     .WORD   10200,0                 ;AC
077062  010000  000000              2$:     .WORD   10000,0                 ;FSRC
077066  000000  000000              3$:     .WORD   0,0                     ;RES
077072  005013                      5$:     5013                            ;FPS BEFORE EXECUTION.
077074  005004                              5004                            ;FPS AFTER EXECUTION.
077076  000012                      6$:     12                              ;FEC
077100  177777                              -1                              ;FLAG
                                    ;OVERFLOW, EXPONENT OF RESULT = 128
077102  004737  077166              III3:   JSR     PC,@#OVUNFNT
077106  060200  000000              1$:     .WORD   60200,0                 ;AC
077112  060000  000000              2$:     .WORD   60000,0                 ;FSRC
077116  000000  000000              3$:     .WORD   0,0                     ;RES
077122  000000                      5$:     0                               ;FPS BEFORE EXECUTION.
```

```
077124  000006                          6                       ;FPS AFTER EXECUTION.
077126  000010                  6$:     10                      ;FEC
077130  000000                          0                       ;FLAG
                                ;OVERFLOW, EXPONENT OF RESULT = 130
077132  004737  077166          III4:   JSR     PC,@#OVUNFNT
077136  060200  000000          1$:     .WORD   60200,0         ;AC
077142  060200  000000          2$:     .WORD   60200,0         ;FSRC
077146  000000  000000          3$:     .WORD   0,0             ;RES
077152  006011                  5$:     6011                    ;FPS BEFORE EXECUTION.
077154  006006                          6006                    ;FPS AFTER EXECUTION.
077156  000010                  6$:     10                      ;FEC
077160  000000                          0                       ;FLAG
077162  000167  000132          8$:     JMP     IIIDONE         ;GO TO NEXT TEST.



                                ;THIS SUBROUTINE, OVUNFNT, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
                                ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
                                ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
                                ;               RES:    .WORD   X,X             ;EXPECTED RESULT
                                ;               ERRES:  .WORD   X,X             ;ERROR RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               FEC:    .WORD   X               ;EXPECTED FEC
                                ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
                                ;               ERR1:   ERROR   X               ;TRAP ERROR.
                                ;               BR      CONT
                                ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE MULF INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
                                ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFNT RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFNT
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFNT
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFNT WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
                                ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNFNT WILL READ THE FEC.
                                ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNFNT WILL
                                ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
                                ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNFNT WILL REPORT
                                ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNFNT USES THE FLAG
                                ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
                                ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

077166  012601                  OVUNFNT:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
077170  012700  000200                  MOV     #200,R0         ;SET FD MODE.
```

```
077174  170100                                LDFPS   R0

077176  010100                                MOV     R1,R0           ;LOAD AC0, OPERAND.
077200  172410                                LDD     (R0),AC0
077202  016100  000014                        MOV     14(R1),R0       ;LOAD THE FPS
077206  170100                                LDFPS   R0
077210  012737  077270  000244                MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
                                                                      ;OF ERROR.
077216  010100                                MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
077220  062700  000004                        ADD     #4,R0

077224  171010                        1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.

077226  170204                        2$:     STFPS   R4              ;GET FPS.
077230  170305                                STST    R5              ;GET FEC.
077232  012700  000200                        MOV     #200,R0         ;SET FD MODE.
077236  170100                                LDFPS   R0
077240  012700  077310                        MOV     #OVFNTT,R0      ;GET THE RESULT.
077244  174010                                STD     AC0,(R0)
077246  012700  077310                        MOV     #OVFNTT,R0      ;CHECK THE RESULT.
077252  010102                                MOV     R1,R2
077254  062702  000010                        ADD     #10,R2
077260  012703  000002                        MOV     #2,R3
077264  022022                        3$:     CMP     (R0)+,(R2)+
077266  001401                                BEQ     5$
077270                                25$:
077270  104000                                EMT                     ;
077272  077304                        5$:     SOB     R3,3$

077274  026104  000016                        CMP     16(R1),R4       ;WAS FPS CORRECT?
077300  001401                                BEQ     4$
077302  104000                                EMT                     ;
077304  000161  000024                4$:     JMP     24(R1)          ;RETURN, TEST COMPLETED.

077310  000000  000000  000000        OVFNTT: .WORD   0,0,0,0
077316  000000

077320                                IIIDONE:
077320  004767  025246                        JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                      ;SEE IF THE USER HAS EXPRESSED
                                                                      ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                      ;THE USER TYPED CONTROL G?).





                                      ;*******************************************************************
                                      ;TEST 502        UNDER\OVER FLOW, USING MULD WITH TRAP DISABLED, TEST
                                      ;*******************************************************************
077324                                TS502:

                                      ;UNDERFLOW, EXPONENT OF RESULT=-129
077324  004737  077550                JJJ1:   JSR     PC,@#OVUNDNT
077330  020200  000000                1$:     .WORD   20200,0         ;AC
077334  127272  000000                        .WORD   127272,0
```

```
077340  020000  000000  000000  2$:     .WORD   20000,0,0,0     ;FSRC
077346  000000
077350  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
077356  000000
077360  000200                  5$:     200                     ;FPS BEFORE EXECUTION.
077362  000204                          204                     ;FPS AFTER EXECUTION.
077364  000012                  6$:     12                      ;FEC
077366  177777                          -1                      ;FLAG

                                ;UNDERFLOW, EXPONENT OF RESULT = -193
077370  004737  077550          JJJ2:   JSR     PC,@#OVUNDNT
077374  010200  000000          1$:     .WORD   10200,0         ;AC
077400  123456  000000                  .WORD   123456,0
077404  010000  000000  000000  2$:     .WORD   10000,0,0,0     ;FSRC
077412  000000
077414  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
077422  000000
077424  005213                  5$:     5213                    ;FPS BEFORE EXECUTION.
077426  005204                          5204                    ;FPS AFTER EXECUTION.
077430  000012                  6$:     12                      ;FEC
077432  177777                          -1                      ;FLAG

                                ;OVERFLOW, EXPONENT OF RESULT = 128
077434  004737  077550          JJJ3:   JSR     PC,@#OVUNDNT
077440  060200  000000          1$:     .WORD   60200,0         ;AC
077444  065432  000000                  .WORD   65432,0
077450  060000  000000  000000  2$:     .WORD   60000,0,0,0     ;FSRC
077456  000000
077460  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
077466  000000
077470  000200                  5$:     200                     ;FPS BEFORE EXECUTION.
077472  000206                          206                     ;FPS AFTER EXECUTION.
077474  000010                  6$:     10                      ;FEC
077476  000000                          0                       ;FLAG

                                ;OVERFLOW, EXPONENT OF RESULT = 130
077500  004737  077550          JJJ4:   JSR     PC,@#OVUNDNT
077504  060200  000000          1$:     .WORD   60200,0         ;AC
077510  125252  000000                  .WORD   125252,0
077514  060200  000000  000000  2$:     .WORD   60200,0,0,0     ;FSRC
077522  000000
077524  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
077532  000000
077534  006211                  5$:     6211                    ;FPS BEFORE EXECUTION.
077536  006206                          6206                    ;FPS AFTER EXECUTION.
077540  000010                  6$:     10                      ;FEC
077542  000000                          0                       ;FLAG
077544  000137  077702          8$:     JMP     @#JJJDONE       ;GO TO NEXT TEST

                                ;THIS SUBROUTINE, OVUNDNT, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
                                ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                ;               FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
```

```
                                  ;               RES:     .WORD    X,X,X,X        ;EXPECTED RESULT
                                  ;               ERRES:   .WORD    X,X,X,X        ;ERROR RESULT
                                  ;               FPSB:    .WORD    X              ;FPS BEFORE EXECUTION
                                  ;               FPSA:    .WORD    X              ;FPS AFTER EXECUTION
                                  ;               FEC:     .WORD    X              ;EXPECTED FEC
                                  ;               FLAG:    .WORD    X              ;0/-1,OVER/UNDER FLOW FLAG
                                  ;               ERR1:    ERROR    X              ;TRAP ERROR.
                                  ;               BR       CONT
                                  ;               ERR2:    ERROR    X              ;DATA, RESULT ERROR
                                  ;               CONT:                            ;RETURN ADDRESS
                                  ;
                                  ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                  ;THE MULD INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
                                  ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                  ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDNT RETURNS CONTROL
                                  ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDNT
                                  ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                  ;MULD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                  ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                  ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDNT
                                  ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
                                  ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDNT WILL
                                  ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
                                  ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNDNT WILL READ THE FEC.
                                  ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNDNT WILL
                                  ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
                                  ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNDNT WILL REPORT
                                  ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNDNT USES THE FLAG
                                  ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
                                  ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

077550  012601                    OVUNDNT:        MOV      (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
077552  012700  000200                    MOV     #200,R0         ;SET FD MODE.
077556  170100                            LDFPS   R0

077560  010100                            MOV     R1,R0           ;LOAD AC0, OPERAND.
077562  172410                            LDD     (R0),AC0
077564  016100  000030                    MOV     30(R1),R0       ;LOAD THE FPS.
077570  170100                            LDFPS   R0
077572  012737  077652  000244            MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
                                                                  ;OF ERROR.
077600  010100                            MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
077602  062700  000010                    ADD     #10,R0

077606  171010                    1$:     MULD    (R0),AC0        ;TEST INSTRUCTION.

077610  170204                    2$:     STFPS   R4              ;GET FPS.
077612  170305                            STST    R5              ;GET FEC.
077614  012700  000200                    MOV     #200,R0         ;SET FD MODE.
077620  170100                            LDFPS   R0
077622  012700  077672                    MOV     #OVDNTT,R0      ;GET THE RESULT.
077626  174010                            STD     AC0,(R0)
077630  012700  077672                    MOV     #OVDNTT,R0      ;CHECK THE RESULT.
077634  010102                            MOV     R1,R2
077636  062702  000020                    ADD     #20,R2
077642  012703  000004                    MOV     #4,R3
```

```
077646  022022                  3$:     CMP     (R0)+,(R2)+
077650  001401                          BEQ     5$
077652                          25$:
077652  104000                          EMT                             ;
077654  077304                  5$:     SOB     R3,3$

077656  026104  000032                  CMP     32(R1),R4               ;WAS FPS CORRECT?
077662  001401                          BEQ     4$
077664  104000                          EMT                             ;
077666  000161  000040          4$:     JMP     40(R1)                  ;RETURN, TEST COMPLETED.

077672  000000  000000  000000  OVDNTT: .WORD   0,0,0,0
077700  000000

077702                          JJJDONE:
077702  004767  024664                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).




                                ;*******************************************************************************
                                ;TEST 503       UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST
                                ;*******************************************************************************
077706                          TS503:

                                ;UNDERFLOW, EXPONENT OF RESULT = -129
077706  004737  100052          KKK1:   JSR     PC,@#OVUNFT
077712  020123  045676          1$:     .WORD   20123,45676             ;AC
077716  020200  000000          2$:     .WORD   20200,0                 ;FSRC
077722  000123  045676          3$:     .WORD   123,45676               ;RES
077726  002000                  5$:     2000                            ;FPS BEFORE EXECUTION.
077730  102004                          102004                          ;FPS AFTER EXECUTION.
077732  000012                  6$:     12                              ;FEC
077734  177777                          -1                              ;FLAG

                                ;UNDERFLOW, EXPONENT OF THE RESULT = -193
077736  004737  100052          KKK3:   JSR     PC,@#OVUNFT
077742  010127  127272          1$:     .WORD   10127,127272            ;AC
077746  010200  000000          2$:     .WORD   10200,0                 ;FSRC
077752  060127  127272          3$:     .WORD   60127,127272            ;RES
077756  007017                  5$:     7017                            ;FPS BEFORE EXECUTION.
077760  107000                          107000                          ;FPS AFTER EXECUTION.
077762  000012                  6$:     12                              ;FEC
077764  177777                          -1

                                ;OVERFLOW, EXPONENT OF THE RESULT = 128
077766  004737  100052          KKK4:   JSR     PC,@#OVUNFT
077772  060252  125252          1$:     .WORD   60252,125252            ;AC
077776  060000  000000          2$:     .WORD   60000,0                 ;FSRC
100002  000052  125252          3$:     .WORD   000052,125252           ;RES
100006  001000                  5$:     1000                            ;FPS BEFORE EXECUTION.
100010  101006                          101006                          ;FPS AFTER EXECUTION.
```

```
100012  000010                  6$:     10                      ;FEC
100014  000000                          0                       ;FLAG

                                ;OVERFLOW, EXPONENT OF RESULT = 130
100016  004737  100052          KKK5:   JSR     PC,@#OVUNFT
100022  060345  067654          1$:     .WORD   60345,67654     ;AC
100026  060200  000000          2$:     .WORD   60200,0         ;FSRC
100032  000345  067654          3$:     .WORD   345,67654       ;RES
100036  007015                  5$:     7015                    ;FPS BEFORE EXECUTION.
100040  107002                          107002                  ;FPS AFTER EXECUTION.
100042  000010                  6$:     10                      ;FEC
100044  000000                          0                       ;FLAG
100046  000167  000162          8$:     JMP     KKKDONE

                                ;THIS SUBROUTINE, OVUNFT, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
                                ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
                                ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
                                ;               RES:    .WORD   X,X             ;EXPECTED RESULT
                                ;               ERRES:  .WORD   X,X             ;ERROR RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               FEC:    .WORD   X               ;EXPECTED FEC
                                ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
                                ;               ERR1:   ERROR   X               ;TRAP ERROR.
                                ;               BR      CONT
                                ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE MULF INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
                                ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFT RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFT
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
                                ;IN THE SAME WAY. IF THE RESULT OF THE
                                ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFT
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFT WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
                                ;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
                                ;NOTE THAT OVUNFNT USES THE FLAG
                                ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
                                ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

100052  012601                  OVUNFT: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
100054  012700  000200                  MOV     #200,R0         ;SET FD MODE.
100060  170100                          LDFPS   R0
100062  010100                          MOV     R1,R0           ;LOAD AC0, OPERAND.
100064  172410                          LDD     (R0),AC0
100066  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS.
```

```
100072  170100                          LDFPS   R0
100074  012737  100116  000244          MOV     #50$,@#FPVECT    ;SET UP THE FP TRAP VECTOR IN CASE
                                                                 ;OF ERROR.
100102  010100                          MOV     R1,R0            ;COMPUTE THE ADDRESS OF FSRC.
100104  062700  000004                  ADD     #4,R0

100110  171010                  1$:     MULF    (R0),AC0         ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
100112  170000                  2$:     CFCC
100114  104000                          EMT                      ;
100116  011602                  50$:    MOV     (SP),R2          ;TRAP TO HERE AND SEE IF THE PC OF THE
100120  020227  100112                  CMP     R2,#2$           ;TRAP WAS THAT OF THE MULF INSTRUCTION.
100124  001401                          BEQ     51$
100126  104000                          EMT                      ;
100130  022626                  51$:    CMP     (SP)+,(SP)+      ;RESET THE STACK
100132  170204                          STFPS   R4               ;GET FPS.
100134  170305                          STST    R5               ;GET FEC.
100136  012700  000200                  MOV     #200,R0          ;SET FD MODE.
100142  170100                          LDFPS   R0
100144  012700  100224                  MOV     #OVFTT,R0        ;GET THE RESULT.
100150  174010                          STD     AC0,(R0)
100152  012700  100224                  MOV     #OVFTT,R0        ;CHECK THE RESULT.
100156  010102                          MOV     R1,R2
100160  062702  000010                  ADD     #10,R2
100164  012703  000002                  MOV     #2,R3
100170  022022                  3$:     CMP     (R0)+,(R2)+
100172  001401                          BEQ     5$
100174  104000                          EMT                      ;
100176  077304                  5$:     SOB     R3,3$

100200  026104  000016                  CMP     16(R1),R4        ;WAS FPS CORRECT?
100204  001401                          BEQ     6$
100206  104000                          EMT                      ;
100210  026105  000020          6$:     CMP     20(R1),R5        ;IS FEC CORRECT?
100214  001401                          BEQ     4$
100216  104000                          EMT                      ;
100220  000161  000024          4$:     JMP     24(R1)           ;RETURN, TEST COMPLETED.

100224  000000  000000  000000  OVFTT:  .WORD   0,0,0,0
100232  000000

100234                          KKKDONE:
100234  004767  024332                  JSR     PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).





                                ;*******************************************************************************
                                ;TEST 504       UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST
                                ;*******************************************************************************
100240                          TS504:

                                ;UNDERFLOW, EXPONENT OF RESULT = -129
```

```
100240  004737  100464          LLL1:   JSR     PC,@#OVUNDT
100244  020052  125252          1$:     .WORD   20052,125252     ;AC
100250  125252  125252                  .WORD   125252,125252
100254  020300  000000  000000  2$:     .WORD   20300,0,0,0      ;FSRC
100262  000000
100264  000177  177777  177777  3$:     .WORD   177,-1,-1,-1     ;RES
100272  177777
100274  002200                  5$:     2200                     ;FPS BEFORE EXECUTION.
100276  102204                          102204                   ;FPS AFTER EXECUTION.
100300  000012                  6$:     12                       ;FEC
100302  177777                          -1                       ;FLAG


                                ;UNDERFLOW, EXPONENT OF THE RESULT = -193
100304  004737  100464          LLL2:   JSR     PC,@#OVUNDT
100310  010327  127272          1$:     .WORD   10327,127272     ;AC
100314  036363  045454                  .WORD   36363,45454
100320  010000  000000  000000  2$:     .WORD   10000,0,0,0      ;FSRC
100326  000000
100330  060127  127272          3$:     .WORD   60127,127272     ;RES
100334  036363  045454                  .WORD   36363,45454
100340  007217                  5$:     7217                     ;FPS BEFORE EXECUTION.
100342  107200                          107200                   ;FPS AFTER EXECUTION.
100344  000012                  6$:     12                       ;FEC
100346  177777                          -1                       ;FLAG


                                ;OVERFLOW, EXPONENT OF THE RESULT = 128
100350  004737  100464          LLL3:   JSR     PC,@#OVUNDT
100354  060252  125252          1$:     .WORD   60252,125252     ;AC
100360  125252  125252                  .WORD   125252,125252    ;FSRC
100364  160100  000000  000000  2$:     .WORD   160100,0,0,0     ;FSRC
100372  000000
100374  100177  177777  177777  3$:     .WORD   100177,-1,-1,-1 ;RES
100402  177777
100404  001200                  5$:     1200                     ;FPS BEFORE EXECUTION.
100406  101216                          101216                   ;FPS AFTER EXECUTION.
100410  000010                  6$:     10                       ;FEC
100412  000000                          0                        ;FLAG


                                ;OVERFLOW, EXPONENT OF THE RESULT = 130
100414  004737  100464          LLL4:   JSR     PC,@#OVUNDT
100420  060345  067654          1$:     .WORD   60345,67654      ;AC
100424  056765  045676                  .WORD   56765,45676
100430  060200  000000  000000  2$:     .WORD   60200,0,0,0      ;FSRC
100436  000000
100440  000345  067654          3$:     .WORD   345,67654        ;RES
100444  056765  045676                  .WORD   56765,45676
100450  007215                  5$:     7215                     ;FPS BEFORE EXECUTION.
100452  107202                          107202                   ;FPS AFTER EXECUTION.
100454  000010                  6$:     10                       ;FEC
100456  000000                          0                        ;FLAG
100460  000137  100646          8$:     JMP     @#LLLDONE

                                ;THIS SUBROUTINE, OVUNDT, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
                                ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
                                ;TO IT IS MADE THUS:
```

```
;
;       ACARG:  .WORD   X,X,X,X         ;AC OPERAND
;       FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
;       RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
;       ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
```

```
                                        ;               FPSA:   .WORD   X                 ;FPS AFTER EXECUTION
                                        ;               FEC:    .WORD   X                 ;EXPECTED FEC
                                        ;               FLAG:   .WORD   X                 ;0/-1,OVER/UNDER FLOW FLAG
                                        ;               ERR1:   ERROR   X                 ;TRAP ERROR.
                                        ;               BR      CONT
                                        ;               ERR2:   ERROR   X                 ;DATA, RESULT ERROR
                                        ;               CONT:                             ;RETURN ADDRESS
                                        ;
                                        ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                        ;THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
                                        ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                        ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
                                        ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
                                        ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
                                        ;IN THE SAME WAY. IF THE RESULT OF THE
                                        ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                        ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                        ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
                                        ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
                                        ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDT WILL
                                        ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
                                        ;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
                                        ;NOTE THAT OVUNDNT USES THE FLAG
                                        ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
                                        ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

100464  012601                          OVUNDT: MOV     (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
100466  012700  000200                          MOV     #200,R0           ;SET FD MODE.
100472  170100                                  LDFPS   R0

100474  010100                                  MOV     R1,R0             ;LOAD AC0, OPERAND.
100476  172410                                  LDD     (R0),AC0
100500  016100  000030                          MOV     30(R1),R0         ;LOAD THE FPS.
100504  170100                                  LDFPS   R0
100506  012737  100530  000244                  MOV     #50$,@#FPVECT     ;SET UP THE FP TRAP VECTOR IN CASE
                                                                          ;OF ERROR.
100514  010100                                  MOV     R1,R0             ;COMPUTE THE ADDRESS OF FSRC.
100516  062700  000010                          ADD     #10,R0

100522  171010                          1$:     MULD    (R0),AC0          ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
100524  170000                          2$:     CFCC
100526  104000                                  EMT                       ;
100530  011602                          50$:    MOV     (SP),R2           ;TRAP TO HERE AND SEE IF THE PC OF THE
100532  020227  100524                          CMP     R2,#2$            ;TRAP WAS THAT OF THE MULF INSTRUCTION.
100536  001401                                  BEQ     51$               ;BRANCH IF YES.
100540  104000                                  EMT                       ;
100542  022626                          51$:    CMP     (SP)+,(SP)+       ;RESET THE STACK
100544  170204                                  STFPS   R4                ;GET FPS.
100546  170305                                  STST    R5                ;GET FEC.
100550  012700  000200                          MOV     #200,R0           ;SET FD MODE.
100554  170100                                  LDFPS   R0
100556  012700  100636                          MOV     #OVDTT,R0         ;GET THE RESULT.
100562  174010                                  STD     AC0,(R0)
100564  012700  100636                          MOV     #OVDTT,R0         ;CHECK THE RESULT.
100570  010102                                  MOV     R1,R2
100572  062702  000020                          ADD     #20,R2
```

```
100576  012703  000004                  MOV     #4,R3
100602  022022                  3$:     CMP     (R0)+,(R2)+
100604  001401                          BEQ     5$
100606  104000                          EMT                     ;
100610  077304                  5$:     SOB     R3,3$
100612  026104  000032                  CMP     32(R1),R4       ;WAS FPS CORRECT?
100616  001401                          BEQ     6$
100620  104000                          EMT                     ;
100622  026105  000034          6$:     CMP     34(R1),R5       ;IS FEC CORRECT?
100626  001401                          BEQ     4$
100630  104000                          EMT                     ;
100632  000161  000040          4$:     JMP     40(R1)          ;RETURN, TEST COMPLETED.

100636  000000  000000  000000  OVDTT:  .WORD   0,0,0,0
100644  000000

100646                          LLLDONE:
100646  004767  023720                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).





                                ;*******************************************************************************
                                ;TEST 505               MODF TEST
                                ;*******************************************************************************
100652                          TS505:

                                ;MODF WITH (FSRC=AC=0)
100652  004737  101376          GGG1:   JSR     PC,@#MODFSUB
100656  000000  000000          1$:     .WORD   0,0             ;AC
100662  000000  000000          2$:     .WORD   0,0             ;FSRC
100666  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
100672  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
100676  000013                  7$:     13                      ;FPS BEFORE EXECUTION.
100700  000004                          4                       ;FPS AFTER EXECUTION.

                                ;MODF TEST, WITH (FSRC=0)
100702  004737  101376          GGG2:   JSR     PC,@#MODFSUB
100706  123456  076543          1$:     .WORD   123456,76543    ;AC
100712  000000  000000          2$:     .WORD   0,0             ;FSRC
100716  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
100722  000000  000000          4$:     .WORD   0,0             ;INTEGER RESULT.
100726  000000                  7$:     0                       ;FPS BEFORE EXECUTION.
100730  000004                          4                       ;FPS AFTER EXECUTION.

                                ;MODF TEST WITH (AC=0)
100732  004737  101376          GGG3:   JSR     PC,@#MODFSUB
100736  000000  000000          1$:     .WORD   0,0             ;AC
100742  076543  021234          2$:     .WORD   76543,21234     ;FSRC
100746  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
100752  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
```

```
100756 000003               7$:     3                       ;FPS BEFORE EXECUTION.
100760 000004                       4                       ;FPS AFTER EXECUTION.

                            ;MODF TEST WITH EXPONENT OF THE RESULT = 25
100762 004737 101376        GGG4:   JSR     PC,@#MODFSUB
100766 046252 125252        1$:     .WORD   46252,125252    ;AC
100772 040300 000000        2$:     .WORD   40300,0         ;FSRC
100776 000000 000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
101002 046377 177777        4$:     .WORD   46377,-1        ;INTEGER RES.
101006 000013               7$:     13                      ;FPS BEFORE EXECUTION.
101010 000004                       4                       ;FPS AFTER EXECUTION.

                            ;MODF TEST WITH EXPONENT OF THE RESULT = 127
101012 004737 101376        GGG5:   JSR     PC,@#MODFSUB
101016 077652 125252        1$:     .WORD   77652,125252    ;AC
101022 040300 000000        2$:     .WORD   40300,0         ;FSRC
101026 000000 000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
101032 077777 177777        4$:     .WORD   77777,-1        ;INTEGER RES.
101036 000000               7$:     0                       ;FPS BEFORE EXECUTION.
101040 000004                       4                       ;FPS AFTER EXECUTION.

                            ;MODF TEST WITH EXPONENT OF RESULT = 25
101042 004737 101376        GGG6:   JSR     PC,@#MODFSUB
101046 046200 000001        1$:     .WORD   46200,1         ;AC
101052 040340 000000        2$:     .WORD   40340,0         ;FSRC
101056 000000 000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
101062 046340 000001        4$:     .WORD   46340,1         ;INTEGER RES.
101066 000013               7$:     13                      ;FPS BEFORE EXECUTION.
101070 000004                       4                       ;FPS AFTER EXECUTION.

                            ;MODF   TEST WITH EXPONENT OF THE RESULT = 24
101072 004737 101376        GGG7:   JSR     PC,@#MODFSUB
101076 046000 000001        1$:     .WORD   46000,1         ;AC
101102 040340 000000        2$:     .WORD   40340,0         ;FSRC
101106 040100 000000        3$:     .WORD   40100,0         ;FRACTIONAL RES.
101112 046140 000001        4$:     .WORD   46140,1         ;INTEGER RESULT.
101116 000000               7$:     0                       ;FPS BEFORE EXECUTION.
101120 000000                       0                       ;FPS AFTER EXECUTION.

                            ;MODF TEST WITH EXPONENT OF THE RESULT = 10
101122 004737 101376        GGG8:   JSR     PC,@#MODFSUB
101126 042577 177777        1$:     .WORD   42577,-1        ;AC
101132 040200 000000        2$:     .WORD   40200,0         ;FSRC
101136 040177 176000        3$:     .WORD   40177,176000    ;FRACTIONAL RES.
101142 042577 140000        4$:     .WORD   42577,140000    ;INTEGER RES.
101146 000000               7$:     0                       ;FPS BEFORE EXECUTION.
101150 000000                       0                       ;FPS AFTER EXECUTION.

                            ;MODF TEST WITH THE EXPONENT OF THE RESULT = 10
101152 004737 101376        GGG9:   JSR     PC,@#MODFSUB
101156 042577 140001        1$:     .WORD   42577,140001    ;AC
101162 040200 000000        2$:     .WORD   40200,0         ;FSRC
101166 034600 000000        3$:     .WORD   34600,0         ;FRACTIONAL RES.
101172 042577 140000        4$:     .WORD   42577,140000    ;INTEGER RES.
101176 000000               7$:     0                       ;FPS BEFORE EXECUTION.
101200 000000                       0                       ;FPS AFTER EXECUTION.
```

```

                                ;MODF TEST WITH EXPONENT OF THE RESULT = 9
101202 004737 101376            GGG10:  JSR     PC,@#MODFSUB
101206 042377 100000            1$:     .WORD   42377,100000    ;AC
101212 040200 000000            2$:     .WORD   40200,0         ;FSRC
101216 000000 000000            3$:     .WORD   0,0             ;FRACTIONAL RES.
101222 042377 100000            4$:     .WORD   42377,100000    ;INTEGER RES.
101226 000013                   7$:     13                      ;FPS BEFORE EXECUTION.
101230 000004                           4                       ;FPS AFTER EXECUTION.

                                ;MODF TEST WITH EXPONENT OF THE RESULT = 0
101232 004737 101376            GGG11:  JSR     PC,@#MODFSUB
101236 040177 177777            1$:     .WORD   40177,-1        ;AC
101242 040200 000000            2$:     .WORD   40200,0         ;FSRC
101246 040177 177777            3$:     .WORD   40177,-1        ;FRACTIONAL RES.
101252 000000 000000            4$:     .WORD   0,0             ;INTEGER RES.
101256 000017                   7$:     17                      ;FPS BEFORE EXECUTION.
101260 000000                           0                       ;FPS AFTER EXECUTION.

                                ;MODF TEST WITH EXPONENT OF THE RESULT = -15
101262 004737 101376            GGG12:  JSR     PC,@#MODFSUB
101266 034377 177777            1$:     .WORD   34377,-1        ;AC
101272 040200 000000            2$:     .WORD   40200,0         ;FSRC
101276 034377 177777            3$:     .WORD   34377,-1        ;FRACTIONAL RES.
101302 000000 000000            4$:     .WORD   0,0             ;INTEGER RES.
101306 000000                   7$:     0                       ;FPS BEFORE EXECUTION.
101310 000000                           0                       ;FPS AFTER EXECUTION.

                                ;MODF TEST WITH EXPONENT OF RESULT = -64, IN ROUND MODE
101312 004737 101376            GGG13:  JSR     PC,@#MODFSUB
101316 020000 000001            1$:     .WORD   20000,1         ;AC
101322 040300 000000            2$:     .WORD   40300,0         ;FSRC
101326 020100 000002            3$:     .WORD   20100,2         ;FRACTIONAL RES.
101332 000000 000000            4$:     .WORD   0,0             ;INTEGER RES.
101336 000000                   7$:     0                       ;FPS BEFORE EXECUTION.
101340 000000                           0                       ;FPS AFTER EXECUTION.

                                ;MODF TEST WITH EXPONENT OF RESULT = 11
101342 004737 101376            GGG14:  JSR     PC,@#MODFSUB
101346 142777 170000            1$:     .WORD   142777,170000   ;AC
101352 040200 000000            2$:     .WORD   40200,0         ;FSRC
101356 140000 000000            3$:     .WORD   140000,0        ;FRACTIONAL RES.
101362 142777 160000            4$:     .WORD   142777,160000   ;INTEGER RES.
101366 000007                   7$:     7                       ;FPS BEFORE EXECUTION.
101370 000010                           10                      ;FPS AFTER EXECUTION.
101372 000167 000204            9$:     JMP     GGGDONE         ;GO TO NEXT TEST.


                                ;THIS SUBROUTINE, MODFSUB, IS CALLED TO SETUP THE
                                ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
                                ;IT IS CALLED THUS:
                                ;
                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
                                ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
                                ;               FRES:   .WORD   X,X             ;FRACTIONAL RESULT
                                ;               INTRES: .WORD   X,X             ;INTEGER RESULT
```

```
                                        ;               ERFRES: .WORD   X,X             ;ERROR FRACTION RESULT
                                        ;               ERINTRES:.WORD  X,X             ;ERROR INTEGER RESULT
                                        ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                        ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                        ;               ERR1:   ERROR   X               ;FRACTION ERROR
                                        ;                       BR      CONT
                                        ;               ERR2:   ERROR   X               ;INTEGER ERROR
                                        ;               CONT:                           ;RETURN ADDRESS
                                        ;
                                        ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
                                        ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
                                        ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
                                        ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
                                        ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
                                        ;THEN MODFSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
                                        ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
                                        ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
                                        ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
                                        ;FAILURE MATCHES THE TRUE RESULT THEN  MODFSUB PASSES CONTROL TO THE
                                        ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
                                        ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
                                        ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
                                        ;IF A MATCH IS MADE HOWEVER, MODFSUB WILL RETURN CONTROL TO THE ERROR
                                        ;CALL AT ERR2.

101376  012601                          MODFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
101400  012700  000200                          MOV     #200,R0                 ;SET FD MODE.
101404  170100                                  LDFPS   R0
101406  010100                                  MOV     R1,R0                   ;SET UP AC0
101410  172410                                  LDD     (R0),AC0
101412  012700  101572                          MOV     #MODP1,R0               ;PUT A BACKROUND PATTERN INTO AC1.
101416  172510                                  LDD     (R0),AC1
101420  016100  000020                          MOV     20(R1),R0               ;SET UP THE FPS.
101424  170100                                  LDFPS   R0
101426  010100                                  MOV     R1,R0                   ;COMPUTE THE ADDRESS OF THE FSRC.
101430  062700  000004                          ADD     #4,R0

101434  171410                          1$:     MODF    (R0),AC0                ;EXECUTE THE TEST INSTRUCTION.

101436  170204                                  STFPS   R4                      ;GET THE FPS.
101440  012700  000200                          MOV     #200,R0                 ;SET FD MODE.
101444  170100                                  LDFPS   R0
101446  012700  101552                          MOV     #MODFT0,R0              ;GET THE FRACTIONAL RESULT.
101452  174010                                  STD     AC0,(R0)
101454  012700  101562                          MOV     #MODFT1,R0              ;GET THE INTEGER RESULT.
101460  174110                                  STD     AC1,(R0)
101462  012702  101552                          MOV     #MODFT0,R2              ;CHECK THE FRACTIONAL RESULT.
101466  026112  000010                          CMP     10(R1),(R2)
101472  001401                                  BEQ     2$
101474  104000                                  EMT                             ;
101476  026162  000012  000002          2$:     CMP     12(R1),2(R2)
101504  001401                                  BEQ     3$
101506  104000                                  EMT                             ;
101510  012702  101562                  3$:     MOV     #MODFT1,R2              ;CHECK THE INTEGER RESULT.
101514  026112  000014                          CMP     14(R1),(R2)
101520  001401                                  BEQ     4$
```

```
101522  104000                           EMT                                  ;
101524  026162  000016  000002  4$:      CMP      16(R1),2(R2)
101532  001401                           BEQ      5$
101534  104000                           EMT                                  ;
101536  026104  000022          5$:      CMP      22(R1),R4                   ;CHECK THE FPS.
101542  001401                           BEQ      9$
101544  104000                           EMT                                  ;
101546  000161  000024          9$:      JMP      24(R1)                      ;RETURN.

101552  000000  000000  000000  MODFT0:  .WORD    0,0,0,0
101560  000000

101562  000000  000000  000000  MODFT1:  .WORD    0,0,0,0
101570  000000

101572  177777  177777  177777  MODP1:   .WORD    -1,-1,-1,-1
101600  177777

101602                          GGGDONE:
101602  004767  022764                   JSR      PC,.RSET                    ;GO INITIALIZE THE FPS AND STACK; AND
                                                                              ;SEE IF THE USER HAS EXPRESSED
                                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                              ;THE USER TYPED CONTROL G?).




                                ;*************************************************************************
                                ;TEST 506        MODD TEST
                                ;*************************************************************************
101606                          TS506:

                                ;MODD WITH (FSRC=AC=0)
101606  004737  102622          HHH1:    JSR      PC,@#MODDSUB
101612  000000  000000  000000  1$:      .WORD    0,0,0,0                     ;AC
101620  000000
101622  000000  000000  000000  2$:      .WORD    0,0,0,0                     ;FSRC
101630  000000
101632  000000  000000  000000  3$:      .WORD    0,0,0,0                     ;FRACTIONAL RES.
101640  000000
101642  000000  000000  000000  4$:      .WORD    0,0,0,0                     ;INTEGER RES.
101650  000000
101652  000200                  7$:      200                                  ;FPS BEFORE EXECUTION.
101654  000204                           204                                  ;FPS AFTER EXECUTION.

                                ;MODD TEST WITH FSRC=0
101656  004737  102622          HHH2:    JSR      PC,@#MODDSUB
101662  012345  067012          1$:      .WORD    012345,67012                ;AC
101666  034567  012345                   .WORD    34567,012345
101672  000000  000000  000000  2$:      .WORD    0,0,0,0                     ;FSRC
101700  000000
101702  000000  000000  000000  3$:      .WORD    0,0,0,0                     ;FRACTIONAL RES.
101710  000000
101712  000000  000000  000000  4$:      .WORD    0,0,0,0                     ;INTEGER RES.
101720  000000
```

```
101722  000213                          7$:     213                             ;FPS BEFORE EXECUTION.
101724  000204                                  204                             ;FPS AFTER EXECUTION.
                                        ;MODD TEST WITH (AC=0)
101726  004737  102622                  HHH3:   JSR     PC,@#MODDSUB
101732  000000  000000  000000          1$:     .WORD   0,0,0,0                 ;AC
101740  000000
101742  072727  127272                  2$:     .WORD   72727,127272            ;FSRC
101746  072727  127272                          .WORD   72727,127272
101752  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;FRACTIONAL RES.
101760  000000
101762  000000  000000  000000          4$:     .WORD   0,0,0,0                 ;INTEGER RES.
101770  000000
101772  000213                          7$:     213                             ;FPS BEFORE EXECUTION.
101774  000204                                  204                             ;FPS AFTER EXECUTION.
                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 57
101776  004737  102622                  HHH4:   JSR     PC,@#MODDSUB
102002  056252  125252                  1$:     .WORD   56252,125252            ;AC
102006  125252  125250                          .WORD   125252,125250
102012  040300  000000  000000          2$:     .WORD   40300,0,0,0             ;FSRC
102020  000000
102022  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;FRACTIONAL RES.
102030  000000
102032  056377  177777  177777          4$:     .WORD   56377,-1,-1,-4          ;INTEGER RES.
102040  177774
102042  000213                          7$:     213                             ;FPS BEFORE EXECUTION.
102044  000204                                  204                             ;FPS AFTER EXECUTION.
                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 79
102046  004737  102622                  HHH5:   JSR     PC,@#MODDSUB
102052  140240  000000  000000          1$:     .WORD   140240,0,0,0            ;AC
102060  000000
102062  063714  146314                  2$:     .WORD   63714,146314            ;FSRC
102066  133572  167737                          .WORD   133572,167737
102072  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;FRACTIONAL RES.
102100  000000
102102  163777  177777                  4$:     .WORD   163777,-1               ;INTEGER RES.
102106  162531  125726                          .WORD   162531,125726
102112  000210                          7$:     210                             ;FPS BEFORE EXECUTION.
102114  000204                                  204                             ;FPS AFTER EXECUTION.
                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 57
102116  004737  102622                  HHH6:   JSR     PC,@#MODDSUB
102122  056200  000000  000000          1$:     .WORD   56200,0,0,1             ;AC
102130  000001
102132  040340  000000  000000          2$:     .WORD   40340,0,0,0             ;FSRC
102140  000000
102142  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;FRACTIONAL RES.
102150  000000
102152  056340  000000  000000          4$:     .WORD   56340,0,0,1             ;INTEGER RES.
102160  000001
102162  000213                          7$:     213                             ;FPS BEFORE EXECUTION.
102164  000204                                  204                             ;FPS AFTER EXECUTION.
                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 56
```

```
102166  004737  102622            HHH7:   JSR     PC,@#MODDSUB
102172  056000  000000  000000    1$:     .WORD   56000,0,0,1     ;AC
102200  000001
102202  040340  000000  000000    2$:     .WORD   40340,0,0,0     ;FSRC
102210  000000
102212  040100  000000  000000    3$:     .WORD   40100,0,0,0     ;FRACTIONAL RES.
102220  000000
102222  056140  000000  000000    4$:     .WORD   56140,0,0,1     ;INTEGER RES.
102230  000001
102232  000213                    7$:     213                     ;FPS BEFORE EXECUTION.
102234  000200                            200                     ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 36
102236  004737  102622            HHH8:   JSR     PC,@#MODDSUB
102242  051177  177777  177777    1$:     .WORD   51177,-1,-1,-1  ;AC
102250  177777
102252  040200  000000  000000    2$:     .WORD   40200,0,0,0     ;FSRC
102260  000000
102262  040177  177760  000000    3$:     .WORD   40177,-20,0,0   ;FRACTIONAL RES.
102270  000000
102272  051177  177777  177760    4$:     .WORD   51177,-1,-20,0  ;INTEGER RES.
102300  000000
102302  000217                    7$:     217                     ;FPS BEFORE EXECUTION.
102304  000200                            200                     ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 30
102306  004737  102622            HHH9:   JSR     PC,@#MODDSUB
102312  040200  000000  000000    1$:     .WORD   40200,0,0,0     ;AC
102320  000000
102322  047577  177777            2$:     .WORD   47577,-1        ;FSRC
102326  176000  000001                    .WORD   176000,1
102332  031600  000000  000000    3$:     .WORD   31600,0,0,0     ;FRACTIONAL RES.
102340  000000
102342  047577  177777            4$:     .WORD   47577,-1        ;INTEGER RES.
102346  176000  000000                    .WORD   176000,0
102352  000200                    7$:     200                     ;FPS BEFORE EXECUTION.
102354  000200                            200                     ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 31
102356  004737  102622            HHH10:  JSR     PC,@#MODDSUB
102362  047777  177777            1$:     .WORD   47777,-1        ;AC
102366  177000  000000                    .WORD   177000,0
102372  040200  000000  000000    2$:     .WORD   40200,0,0,0     ;FSRC
102400  000000
102402  000000  000000  000000    3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
102410  000000
102412  047777  177777            4$:     .WORD   47777,-1        ;INTEGER RES.
102416  177000  000000                    .WORD   177000,0
102422  000213                    7$:     213                     ;FPS BEFORE EXECUTION.
102424  000204                            204                     ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 0
102426  004737  102622            HHH11:  JSR     PC,@#MODDSUB
102432  040200  000000  000000    1$:     .WORD   40200,0,0,0     ;AC
102440  000000
102442  040177  072727            2$:     .WORD   40177,72727     ;FSRC
```

```
102446  127272  072727                    .WORD   127272,72727
102452  040177  072727            3$:     .WORD   40177,72727      ;FRACTIONAL RES.
102456  127272  072727                    .WORD   127272,72727
102462  000000  000000  000000    4$:     .WORD   0,0,0,0          ;INTEGER RES.
102470  000000
102472  000200                    7$:     200                      ;FPS BEFORE EXECUTION.
102474  000200                            200                      ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -115
102476  004737  102622            HHH12:  JSR     PC,@#MODDSUB
102502  003377  177777            1$:     .WORD   3377,-1          ;AC
102506  177777  052525                    .WORD   -1,52525
102512  040200  000000  000000    2$:     .WORD   40200,0,0,0      ;FSRC
102520  000000
102522  003377  177777            3$:     .WORD   3377,-1          ;FRACTIONAL RES.
102526  177777  052525                    .WORD   -1,52525
102532  000000  000000  000000    4$:     .WORD   0,0,0,0          ;INTEGER RES.
102540  000000
102542  000200                    7$:     200                      ;FPS BEFORE EXECUTION.
102544  000200                            200                      ;FPS AFTER EXECUTION.

                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.
102546  004737  102622            HHH13:  JSR     PC,@#MODDSUB
102552  040300  000000  000000    1$:     .WORD   40300,0,0,0      ;AC
102560  000000
102562  020200  000000  000000    2$:     .WORD   20200,0,0,1      ;FSRC
102570  000001
102572  020300  000000  000000    3$:     .WORD   20300,0,0,2      ;FRACTIONAL RES.
102600  000002
102602  000000  000000  000000    4$:     .WORD   0,0,0,0          ;INTEGER RES.
102610  000000
102612  000200                    7$:     200                      ;FPS BEFORE EXECUTION.
102614  000200                            200                      ;FPS AFTER EXECUTION.
102616  000137  103016            9$:     JMP     @#HHHDONE        ;GO TO THE NEXT TEST.

                                  ;THIS SUBROUTINE, MODDSUB, IS CALLED TO SETUP THE
                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
                                  ;IT IS CALLED THUS:
                                  ;
                                  ;       ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                  ;       FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
                                  ;       FRES:   .WORD   X,X,X,X         ;FRACTIONAL RESULT
                                  ;       INTRES: .WORD   X,X,X,X         ;INTEGER RESULT
                                  ;       ERFRES: .WORD   X,X,X,X         ;ERROR FRACTION RESULT
                                  ;       ERINTRES:.WORD  X,X,X,X         ;ERROR INTEGER RESULT
                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                  ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                  ;       ERR1:   ERROR   X               ;FRACTION ERROR
                                  ;               BR      CONT
                                  ;       ERR2:   ERROR   X               ;INTEGER ERROR
                                  ;       CONT:                           ;RETURN ADDRESS
                                  ;
                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
```

```
                                        ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
                                        ;THEN MODDSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
                                        ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
                                        ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
                                        ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
                                        ;FAILURE MATCHES THE TRUE RESULT THEN  MODDSUB PASSES CONTROL TO THE
                                        ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
                                        ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
                                        ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
                                        ;IF A MATCH IS MADE HOWEVER, MODDSUB WILL RETURN CONTROL TO THE ERROR
                                        ;CALL AT ERR2.

102622  012601                          MODDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
102624  012700  000200                          MOV     #200,R0         ;SET FD MODE.
102630  170100                                  LDFPS   R0
102632  010100                                  MOV     R1,R0           ;SET UP AC0
102634  172410                                  LDD     (R0),AC0
102636  012700  101572                          MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
102642  172510                                  LDD     (R0),AC1
102644  016100  000040                          MOV     40(R1),R0       ;SET UP THE FPS.
102650  170100                                  LDFPS   R0
102652  010100                                  MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
102654  062700  000010                          ADD     #10,R0

102660  171410                          1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

102662  170204                                  STFPS   R4              ;GET THE FPS.
102664  012700  000200                          MOV     #200,R0         ;SET FD MODE.
102670  170100                                  LDFPS   R0
102672  012700  102776                          MOV     #MODDT0,R0      ;GET THE FRACTIONAL RESULT.
102676  174010                                  STD     AC0,(R0)
102700  012700  103006                          MOV     #MODDT1,R0      ;GET THE INTEGER RESULT.
102704  174110                                  STD     AC1,(R0)
102706  012702  102776                          MOV     #MODDT0,R2      ;CHECK THE FRACTIONAL RESULT.
102712  010103                                  MOV     R1,R3
102714  062703  000020                          ADD     #20,R3
102720  012705  000004                          MOV     #4,R5
102724  022223                          2$:     CMP     (R2)+,(R3)+
102726  001401                                  BEQ     4$
102730  104000                                  EMT                     ;
102732  077504                          4$:     SOB     R5,2$
102734  012702  103006                          MOV     #MODDT1,R2      ;CHECK THE INTEGER RESULT.
102740  010103                                  MOV     R1,R3
102742  062703  000030                          ADD     #30,R3
102746  012705  000004                          MOV     #4,R5
102752  022223                          3$:     CMP     (R2)+,(R3)+
102754  001401                                  BEQ     5$
102756  104000                                  EMT                     ;
102760  077504                          5$:     SOB     R5,3$
102762  026104  000042                          CMP     42(R1),R4       ;CHECK THE FPS.
102766  001401                                  BEQ     9$
102770  104000                                  EMT                     ;
102772  000161  000044                  9$:     JMP     44(R1)          ;RETURN.

102776  000000  000000  000000  MODDT0: .WORD   0,0,0,0
103004  000000
```

```
103006  000000  000000  000000  MODDT1: .WORD    0,0,0,0
103014  000000

103016                          HHHDONE:
103016  004767  021550                  JSR      PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 507        UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST
                                ;*******************************************************************************
103022                          TS507:

                                ;UNDERFLOW TEST, WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
103022  004767  000214          MMM1:   JSR      PC,MODFOV
103026  020123  045676          1$:     .WORD    20123,45676     ;AC
103032  020200  000000          2$:     .WORD    20200,0         ;FSRC
103036  000123  045676          3$:     .WORD    123,45676       ;FRACTIONAL RES.
103042  000000  000000          4$:     .WORD    0,0             ;INTEGER RES.
103046  042000                  7$:     42000                    ;FPS BEFORE EXECUTION.
103050  142004                          142004                   ;FPS AFTER EXECUTION.
103052  000012                          12                       ;FEC
103054  104000                          EMT                      ;

                                ;UNDERFLOW EXP OF RESULT = -193, FIU = 0, FID = 1
103056  004737  103242          MMM2:   JSR      PC,@#MODFOV
103062  010200  000000          1$:     .WORD    10200,0         ;AC
103066  010000  000000          2$:     .WORD    10000,0         ;FSRC
103072  000000  000000          3$:     .WORD    0,0             ;FRACTIONAL RES.
103076  000000  000000          4$:     .WORD    0,0             ;INTEGER RES.
103102  005013                  7$:     5013                     ;FPS BEFORE EXECUTION.
103104  005004                          5004                     ;FPS AFTER EXECUTION.
103106  000012                          12                       ;FEC
103110  000240                          NOP

                                ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
103112  004737  103242          MMM3:   JSR      PC,@#MODFOV
103116  060052  125252          1$:     .WORD    60052,125252    ;AC
103122  060200  000000          2$:     .WORD    60200,0         ;FSRC
103126  000000  000000          3$:     .WORD    0,0             ;FRACTIONAL RES.
103132  000052  125252          4$:     .WORD    52,125252       ;INTEGER RES.
103136  041000                  7$:     41000                    ;FPS BEFORE EXECUTION.
103140  141006                          141006                   ;FPS AFTER EXECUTION.
103142  000010                          10                       ;FEC
103144                          8$:
103144  104000                          EMT                      ;
                                ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
103146  004737  103242          MMM4:   JSR      PC,@#MODFOV
103152  060345  067654          1$:     .WORD    60345,67654     ;AC
103156  060200  000000          2$:     .WORD    60200,0         ;FSRC
```

```
103162  000000  000000          3$:     .WORD   0,0                     ;FRACTIONAL RES.
103166  000000  000000          4$:     .WORD   0,0                     ;INTEGER RES.
103172  006011                  7$:     6011                            ;FPS BEFORE EXECUTION.
103174  006006                          6006                            ;FPS AFTER EXECUTION.
103176  000010                          10                              ;FEC
103200  000240                  8$:     NOP
                                ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
                                ;AND FIV = 1, FID = 1
103202  004737  103242          MMM5:   JSR     PC,@#MODFOV
103206  160252  125252          1$:     .WORD   160252,125252           ;AC
103212  060000  000000          2$:     .WORD   60000,0                 ;FSRC
103216  000000  000000          3$:     .WORD   0,0                     ;FRACTIONAL RES.
103222  100052  125252          4$:     .WORD   100052,125252           ;INTEGER RES.
103226  041000                  7$:     41000                           ;FPS BEFORE EXECUTION.
103230  141006                          141006                          ;FPS AFTER EXECUTION.
103232  000010                          10                              ;FEC
103234                          8$:
103234  104000                          EMT                             ;
103236  000137  103456          9$:     JMP     @#MMMDONE               ;GO TO THE NEXT TEST.

                                ;THIS SUBROUTINE, MODFOV, IS CALLED TO SETUP THE
                                ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
                                ;IT IS CALLED THUS:
                                ;
                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
                                ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
                                ;               FRES:   .WORD   X,X             ;FRACTIONAL RESULT
                                ;               INTRES: .WORD   X,X             ;INTEGER RESULT
                                ;               ERFRES: .WORD   X,X             ;ERROR FRACTION RESULT
                                ;               ERINTRES:.WORD  X,X             ;ERROR INTEGER RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               FEC:    .WORD   X               ;FEC
                                ;               ERR1:   ERROR   X               ;FEC ERROR
                                ;                       BR      CONT
                                ;               ERR2:   ERROR   X               ;INTEGER ERROR
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
                                ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
                                ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
                                ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
                                ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
                                ;THEN MODFOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
                                ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
                                ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
                                ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
                                ;FAILURE MATCHES THE TRUE RESULT THEN  MODFOV PASSES CONTROL TO THE
                                ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
                                ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
                                ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
                                ;IF A MATCH IS MADE HOWEVER, MODFOV WILL RETURN CONTROL TO THE ERROR
                                ;CALL AT ERR2.

103242  012601                  MODFOV: MOV     (SP)+,R1                ;GET A POINTER TO THE ARGUMENTS
103244' 012700  000200                  MOV     #200,R0                 ;SET FD MODE.
```

```
103250  170100                            LDFPS   R0
103252  010100                            MOV     R1,R0           ;SET UP AC0
103254  172410                            LDD     (R0),AC0
103256  012700  101572                    MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
103262  172510                            LDD     (R0),AC1
103264  016100  000020                    MOV     20(R1),R0       ;SET UP THE FPS.
103270  170100                            LDFPS   R0
103272  010100                            MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
103274  062700  000004                    ADD     #4,R0

103300  171410                    1$:     MODF    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

103302  170204                            STFPS   R4              ;GET THE FPS.
103304  170305                            STST    R5              ;GET FEC.
103306  012700  000200                    MOV     #200,R0         ;SET FD MODE.
103312  170100                            LDFPS   R0
103314  012700  103436                    MOV     #MODFD0,R0      ;GET THE FRACTIONAL RESULT.
103320  174010                            STD     AC0,(R0)
103322  012700  103446                    MOV     #MODFD1,R0      ;GET THE INTEGER RESULT.
103326  174110                            STD     AC1,(R0)
103330  012702  103436                    MOV     #MODFD0,R2      ;CHECK THE FRACTIONAL RESULT.
103334  026112  000010                    CMP     10(R1),(R2)
103340  001401                            BEQ     2$
103342  104000                            EMT                     ;
103344  026162  000012  000002    2$:     CMP     12(R1),2(R2)
103352  001401                            BEQ     3$
103354  104000                            EMT                     ;
103356  012702  103446            3$:     MOV     #MODFD1,R2      ;CHECK THE INTEGER RESULT.
103362  026112  000014                    CMP     14(R1),(R2)
103366  001401                            BEQ     4$
103370  104000                            EMT                     ;
103372  026162  000016  000002    4$:     CMP     16(R1),2(R2)
103400  001401                            BEQ     5$
103402  104000                            EMT                     ;
103404  026104  000022            5$:     CMP     22(R1),R4       ;CHECK THE FPS.
103410  001401                            BEQ     6$
103412  104000                            EMT                     ;
103414  026105  000024            6$:     CMP     24(R1),R5       ;CHECK THE FEC.
103420  001002                            BNE     25$             ;BRANCH IF INCORRECT.

103422  000161  000030            9$:     JMP     30(R1)          ;RETURN.
                                  ;REPORT FEC ERROR.
103426  010102                    25$:    MOV     R1,R2
103430  062702  000026                    ADD     #26,R2
103434  000112                            JMP     (R2)

103436  000000  000000  000000    MODFD0: .WORD   0,0,0,0
103444  000000

103446  000000  000000  000000    MODFD1: .WORD   0,0,0,0
103454  000000

103456                            MMMDONE:
103456  004767  021110                    JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                  ;SEE IF THE USER HAS EXPRESSED
                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
```

```
                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                  ;THE USER TYPED CONTROL G?).




                                        ;*************************************************************************
                                        ;TEST 510        UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST
                                        ;*************************************************************************
103462                                  TS510:

                                        ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
103462  004737  103746                  NNN1:   JSR     PC,@#MODDOV
103466  020252  125252                  1$:     .WORD   20252,125252    ;AC
103472  125252  125252                          .WORD   125252,125252
103476  020100  000000  000000          2$:     .WORD   20100,0,0,0     ;FSRC
103504  000000
103506  000177  177777  177777          3$:     .WORD   177,-1,-1,-1    ;FRACTIONAL RES.
103514  177777
103516  000000  000000  000000          4$:     .WORD   0,0,0,0         ;INTEGER RES.
103524  000000
103526  042200                          7$:     42200                   ;FPS BEFORE EXECUTION.
103530  142204                                  142204                  ;FPS AFTER EXECUTION.
103532  000012                                  12                      ;FEC
103534                                  8$:
103534  104000                                  EMT                     ;
                                        ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -193, FIU = 0, FID = 1
103536  004737  103746                  NNN2:   JSR     PC,@#MODDOV
103542  010000  000000                  1$:     .WORD   10000,0         ;AC
103546  123456  000000                          .WORD   123456,0
103552  010200  000000  000000          2$:     .WORD   10200,0,0,0     ;FSRC
103560  000000
103562  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
103570  000000
103572  000000  000000  000000          4$:     .WORD   0,0,0,0         ;INTEGER RES.
103600  000000
103602  005213                          7$:     5213                    ;FPS BEFORE EXECUTION.
103604  005204                                  5204                    ;FPS AFTER EXECUTION.
103606  000012                                  12
103610  000240                          8$:     NOP
                                        ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
103612  004737  103746                  NNN3:   JSR     PC,@#MODDOV
103616  060252  125252                  1$:     .WORD   60252,125252    ;AC
103622  125252  125252                          .WORD   125252,125252
103626  060100  000000  000000          2$:     .WORD   60100,0,0,0     ;FSRC
103634  000000
103636  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
103644  000000
103646  000177  177777  177777          4$:     .WORD   177,-1,-1,-1    ;INTEGER RES.
103654  177777
103656  041200                          7$:     41200                   ;FPS BEFORE EXECUTION.
103660  141206                                  141206                  ;FPS AFTER EXECUTION.
103662  000010                                  10                      ;FEC
103664                                  8$:
103664  104000                                  EMT                     ;
                                        ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
```

```
103666  004737  103746            NNN4:   JSR     PC,@#MODDOV
103672  060200  000000            1$:     .WORD   60200,0         ;AC
103676  125252  000000                    .WORD   125252,0
103702  060200  000000  000000    2$:     .WORD   60200,0,0,0     ;FSRC
103710  000000
103712  000000  000000  000000    3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
103720  000000
103722  000000  000000  000000    4$:     .WORD   0,0,0,0         ;INTEGER RES.
103730  000000
103732  006211                    7$:     6211                    ;FPS BEFORE EXECUTION.
103734  006206                            6206                    ;FPS AFTER EXECUTION.
103736  000010                            10                      ;FEC
103740  000240                    8$:     NOP
103742  000137  104162            9$:     JMP     @#NNNDONE       ;GO TO NEXT TEST.

                                  ;THIS SUBROUTINE, MODDOV, IS CALLED TO SETUP THE
                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
                                  ;IT IS CALLED THUS:
                                  ;
                                  ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                  ;               FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
                                  ;               FRES:   .WORD   X,X,X,X         ;FRACTIONAL RESULT
                                  ;               INTRES: .WORD   X,X,X,X         ;INTEGER RESULT
                                  ;               ERFRES: .WORD   X,X,X,X         ;ERROR FRACTION RESULT
                                  ;               ERINTRES:.WORD  X,X,X,X         ;ERROR INTEGER RESULT
                                  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                  ;               ERR1:   ERROR   X               ;FRACTION ERROR
                                  ;                       BR      CONT
                                  ;               ERR2:   ERROR   X               ;INTEGER ERROR
                                  ;               CONT:                           ;RETURN ADDRESS
                                  ;
                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
                                  ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
                                  ;THEN MODDOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODDOV PASSES CONTROL TO THE
                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
                                  ;IF A MATCH IS MADE HOWEVER, MODDOV WILL RETURN CONTROL TO THE ERROR
                                  ;CALL AT ERR2.

103746  012601                    MODDOV: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
103750  012700  000200                    MOV     #200,R0         ;SET FD MODE.
103754  170100                            LDFPS   R0
103756  010100                            MOV     R1,R0           ;SET UP AC0
103760  172410                            LDD     (R0),AC0
103762  012700  101572                    MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
103766  172510                            LDD     (R0),AC1
103770  016100  000040                    MOV     40(R1),R0       ;SET UP THE FPS.
```

```
103774  170100                              LDFPS   R0
103776  010100                              MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
104000  062700  000010                      ADD     #10,R0

104004  171410                      1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

104006  170305                              STST    R5              ;GET THE FPS.
104010  170204                              STFPS   R4              ;GET THE FPS.
104012  012700  000200                      MOV     #200,R0         ;SET FD MODE.
104016  170100                              LDFPS   R0
104020  012700  104142                      MOV     #MODDD0,R0      ;GET THE FRACTIONAL RESULT.
104024  174010                              STD     AC0,(R0)
104026  012700  104152                      MOV     #MODDD1,R0      ;GET THE INTEGER RESULT.
104032  174110                              STD     AC1,(R0)
104034  012702  104142                      MOV     #MODDD0,R2      ;CHECK THE FRACTIONAL RESULT.
104040  010103                              MOV     R1,R3
104042  062703  000020                      ADD     #20,R3
104046  012700  000004                      MOV     #4,R0
104052  022223                      2$:     CMP     (R2)+,(R3)+
104054  001401                              BEQ     4$
104056  104000                              EMT                     ;
104060  077004                      4$:     SOB     R0,2$
104062  012702  104152                      MOV     #MODDD1,R2      ;CHECK THE INTEGER RESULT
104066  010103                              MOV     R1,R3
104070  062703  000030                      ADD     #30,R3
104074  012700  000004                      MOV     #4,R0
104100  022223                      3$:     CMP     (R2)+,(R3)+
104102  001401                              BEQ     5$
104104  104000                              EMT                     ;
104106  077004                      5$:     SOB     R0,3$
104110  026104  000042                      CMP     42(R1),R4       ;CHECK THE FPS.
104114  001401                              BEQ     6$
104116  104000                              EMT                     ;
104120  026105  000044              6$:     CMP     44(R1),R5       ;CHECK THE FEC.
104124  001002                              BNE     25$

104126  000161  000050              9$:     JMP     50(R1)          ;RETURN.
                                    ;REPORT FEC ERROR.
104132  010102                      25$:    MOV     R1,R2
104134  062702  000046                      ADD     #46,R2
104140  000112                              JMP     (R2)

104142  000000  000000  000000      MODDD0: .WORD   0,0,0,0
104150  000000

104152  000000  000000  000000      MODDD1: .WORD   0,0,0,0
104160  000000

104162                              NNNDONE:
104162  004767  020404                      JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                    ;SEE IF THE USER HAS EXPRESSED
                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                    ;THE USER TYPED CONTROL G?).
```

```
                                        ;*************************************************************************
                                        ;TEST 511       MORE MICROCODES COVERAGE
                                        ;*************************************************************************
104166                                  TS511:
104166  012737  104202  000244          XT1:    MOV     #XT1A,@#244
104174  170227  000000                          STFPS   #0
104200  000401                                  BR      XT2
104202                                  XT1A:
104202  104000                                  EMT                             ;

104204  012700  177777                  XT2:    MOV     #-1,R0
104210  170127  000000                          LDFPS   #0
104214  170200                                  STFPS   R0
104216  005700                                  TST     R0
104220  001401                                  BEQ     XT2A
104222  104000                                  EMT                             ;
104224  012700  104760                  XT2A:   MOV     #XPAT0,R0
104230  172440                                  LDF     -(R0),AC0
104232  022700  104754                          CMP     #XPAT0-4,R0
104236  001401                                  BEQ     XT2B
104240  104000                                  EMT                             ;
104242  170200                          XT2B:   STFPS   R0
104244  022700  000004                          CMP     #4,R0                   ;CHECK IF FZ IS SET?
104250  001401                                  BEQ     XT3
104252  104000                                  EMT                             ;
104254  170127  000000                  XT3:    LDFPS   #0
104260  012700  104760                          MOV     #XPAT0,R0
104264  174040                                  STF     AC0,-(R0)
104266  022700  104754                          CMP     #XPAT0-4,R0
104272  001401                                  BEQ     XT3A
104274  104000                                  EMT                             ;
104276  170200                          XT3A:   STFPS   R0
104300  005700                                  TST     R0
104302  001401                                  BEQ     XT4
104304  104000                                  EMT                             ;

104306  170127  000000                  XT4:    LDFPS   #0
104312  012737  104336  000244                  MOV     #XT4A,@#244
104320  170127  004000                          LDFPS   #04000                  ;INTRPT ON UNDEFINED VARIABLE
104324  172437  104760                          LDF     @#XPAT0,AC0
104330  174437  105010                          DIVF    @#XPAT3,AC0             ;GET UNDEFINED VARIABLE, _0
104334  104000                                  EMT                             ;
104336  170200                          XT4A:   STFPS   R0
104340  022700  104004                          CMP     #104004,R0              ;CHECK: FER,FIUV,FZ ARE SET?
104344  001401                                  BEQ     XT4B
104346  104000                                  EMT                             ;
104350  012700  104750                  XT4B:   MOV     #XBUF,R0
104354  174010                                  STF     AC0,(R0)
104356  005737  104750                          TST     @#XBUF
104362  001401                                  BEQ     XT5
104364  104000                                  EMT                             ;

104366  012737  104406  000244          XT5:    MOV     #XT5A,@#244
104374  170127  004000                          LDFPS   #04000                  ;INTRPT ON UNDEFINED VARIBALE
104400  177437  105010                          LDCDF   @#XPAT3,AC0             ;GET UNDEFINED VARIABLE, _0
104404  104000                                  EMT                             ;
```

```
104406  170200                        XT5A:   STFPS   R0
104410  022700  104014                        CMP     #104014,R0         ;CHECK: FER,FIUV,FN,FZ ARE SET?
104414  001401                                BEQ     XT5B
104416  104000                                EMT                        ;
104420  012700  104750                XT5B:   MOV     #XBUF,R0
104424  174010                                STF     AC0,(R0)
104426  005737  104750                        TST     @#XBUF
104432  001401                                BEQ     XT6
104434  104000                                EMT                        ;

104436  012737  104462  000244        XT6:    MOV     #XT6A,@#244
104444  170127  004000                        LDFPS   #04000             ;INTRPT ON UNDEFINED VARIBALE
104450  172437  104760                        LDF     @#XPAT0,AC0
104454  172037  105010                        ADDF    @#XPAT3,AC0
104460  104000                                EMT                        ;
104462  170200                        XT6A:   STFPS   R0
104464  022700  104004                        CMP     #104004,R0         ;CHECK: FER,FIUV,FZ ARE SET?
104470  001401                                BEQ     XT6B
104472  104000                                EMT                        ;
104474  012700  104750                XT6B:   MOV     #XBUF,R0
104500  174010                                STF     AC0,(R0)
104502  005737  104750                        TST     @#XBUF
104506  001401                                BEQ     XT7
104510  104000                                EMT                        ;

104512  170127  000000                XT7:    LDFPS   #0
104516  172437  105020                        LDF     @#XPAT4,AC0
104522  175437  105050                        STCFI   AC0,@#XPAT0
104526  022737  000002  105050                CMP     #2,@#XPAT0         ;CHECK DATA
104534  001401                                BEQ     XT8
104536  104000                                EMT                        ;

104540  170127  000100                XT8:    LDFPS   #100               ;SET FL
104544  172437  105020                        LDF     @#XPAT4,AC0
104550  175467  000274                        STCFI   AC0,XPAT0
104554  022737  000002  105052                CMP     #2,@#XPAT0+2
104562  001401                                BEQ     XT9
104564  104000                                EMT                        ;

                                      ;START OF FPP2
104566  170127  000000                XT9:    LDFPS   #0
104572  172437  104760                        LDF     @#XPAT0,AC0
104576  172037  105020                        ADDF    @#XPAT4,AC0
104602  170200                                STFPS   R0
104604  005700                                TST     R0
104606  001401                                BEQ     XT10
104610  104000                                EMT                        ;

104612  170127  000000                XT10:   LDFPS   #0
104616  172437  105020                        LDF     @#XPAT4,AC0
104622  173037  105020                        SUBF    @#XPAT4,AC0
104626  170200                                STFPS   R0
104630  022700  000004                        CMP     #4,R0
104634  001401                                BEQ     XT11
104636  104000                                EMT                        ;

```

```
104640 170127 000000         XT11:    LDFPS   #0
104644 172437 105020                  LDF     @#XPAT4,AC0
104650 173437 105020                  CMPF    @#XPAT4,AC0
104654 170200                         STFPS   R0
104656 022700 000004                  CMP     #4,R0              ;CHECK IF FZ IS SET?
104662 001401                         BEQ     XT12
104664 104000                         EMT                        ;

104666 170127 000000         XT12:    LDFPS   #0
104672 172437 105020                  LDF     @#XPAT4,AC0
104676 174437 105000                  DIVF    @#XPAT2,AC0
104702 012700 104750                  MOV     #XBUF,R0
104706 174010                         STF     AC0,(R0)
104710 022737 040176 104750           CMP     #040176,@#XBUF     ;CHECK DATA
104716 001401                         BEQ     XT13
104720 104000                         EMT                        ;

104722 170127 000000         XT13:    LDFPS   #0
104726 172437 105030                  LDF     @#XPAT5,AC0
104732 174437 105040                  DIVF    @#XPAT6,AC0
104736 170200                         STFPS   R0
104740 022700 000004                  CMP     #4,R0
104744 001445                         BEQ     XTDONE
104746 104000                         EMT                        ;


104750 000000 000000 000000  XBUF:    .WORD   0,0,0,0
104756 000000
104760 000000 000000 000000  XPAT0:   .WORD   0,0,0,0
104766 000000
104770 000001 000001 000001  XPAT1:   .WORD   1,1,1,1
104776 000001
105000 040401 000000 000000  XPAT2:   .WORD   40401,0,0,0
105006 000000
105010 100000 000000 000000  XPAT3:   .WORD   100000,0,0,0
105016 000000
105020 040400 000000 000000  XPAT4:   .WORD   040400,0,0,0
105026 000000
105030 000207 000000 000000  XPAT5:   .WORD   207,0,0,0
105036 000000
105040 077007 000000 000000  XPAT6:   .WORD   77007,0,0,0
105046 000000
105050 000000 000000 000000  XPAT0:   .WORD   0,0,0,0
105056 000000

105060                       XTDONE:
105060 004767 017506                  JSR     PC,.RSET           ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).




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

```
                                        ;TEST 512       STF WITH ILLEGAL ACCUMULATOR TEST
                                        ;*******************************************************************************
105064                                  TS512:

105064  005000                                  CLR     R0                      ;SET THE FPS.
105066  170100                                  LDFPS   R0

105070  012737  105110  000244                  MOV     #000T,@#FPVECT          ;SET UP FOR FP TRAPS.
105076  012737  105104  037364                  MOV     #1$,@#$TMP2

105104  174007                          1$:     STF     AC0,AC7                 ;THIS TEST INSTRUCTION SHOULD
                                                                                ;CAUSE A TRAP.

                                        ;REPORT FAILURE OF USE OF ILLEGAL ACCUMULATOR 7 TO CAUSE AN FPP TRAP.
105106                                  0002:
105106  104000                                  EMT                             ;INSTRUCTION DID NOT TRAP

                                        ;TRAP TO 000T, HERE, WHEN THE EXPECTED ERROR OCCURS.
105110  011600                          000T:   MOV     (SP),R0                 ;MAKE SURE THE ERROR OCCURRED
105112  022700  105106                          CMP     #0002,R0                ;AT THE CORRECT ADDRESS.
105116  001420                                  BEQ     TS513
105120  104000                                  EMT                             ;FLOATING POINT TRAP DID NOT OPERATE RIGHT

105122  170204                          0003:   STFPS   R4                      ;GET FPS.
105124  170305                                  STST    R5                      ;GET FEC.
105126  012702  100000                          MOV     #100000,R2              ;EXPECTED FPS
105132  012703  000002                          MOV     #2,R3                   ;EXPECTED FEC
105136  022626                                  CMP     (SP)+,(SP)+             ;RESET THE STACK.

105140  020204                                  CMP     R2,R4                   ;WAS FPS CORRECT?
105142  001401                                  BEQ     0004
105144  104000                                  EMT                             ;FPS INCORRECTLY SET AFTER USE OF ILLEGAL ACC
105146  020305                          0004:   CMP     R3,R5                   ;WAS THE FEC CORRECT?
105150  001401                                  BEQ     000DONE
105152  104000                                  EMT                             ;INCORRECT FEC AFTER USE OF ILLEGAL ACC

105154                                  000DONE:
105154  004767  017412                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).





                                        ;*******************************************************************************
                                        ;TEST 513       FDST MODE 1, FLOATING MODE, TEST
                                        ;*******************************************************************************
105160                                  TS513:


105160  012700  177777                          MOV     #-1,R0                  ;SET UP A BACKROUND PATTERN IN THE
105164  012701  105274                          MOV     #PPPBF0,R1              ;INPUT BUFFER.
105170  012702  000014                          MOV     #14,R2
105174  010021                          PPP2:   MOV     R0,(R1)+
```

```
18616 105176 077202                 SOB     R2,PPP2
18617
18618 105200 012700 000200          MOV     #200,R0         ;SET FD MODE.
18619 105204 170100                 LDFPS   R0
18620 105206 012700 105324          MOV     #PPPTP1,R0      ;PUT TEST DATA INTO AC0.
18621 105212 172410                 LDD     (R0),AC0
18622
```

```
105214  012700  105310                    MOV    #PPPBF1,R0       ;FDST ADDRESS.
105220  005002                            CLR    R2               ;CLEAR THE FPS.
105222  170102                            LDFPS  R2

105224  174010                    PPP3:   STF    AC0,(R0)         ;TEST INSTRUCTION.

105226  022700  105310                    CMP    #PPPBF1,R0       ;WAS R0 MODIFIED DURING EXECUTION?
105232  001401                            BEQ    PPP4
105234  104000                            EMT                     ;R0 MODIFIED

105236  012700  105310            PPP4:   MOV    #PPPBF1,R0       ;CHECK THE DATA IN THE OUTPUT BUFFER.
105242  012701  105324                    MOV    #PPPTP1,R1
105246  022021                            CMP    (R0)+,(R1)+
105250  001031                            BNE    PPP10            ;BRANCH IF INCORRECT.
105252  022011                            CMP    (R0)+,(R1)
105254  001027                            BNE    PPP10            ;BRANCH IF INCORRECT.
105256  022720  177777                    CMP    #-1,(R0)+        ;WAS FLOATING MODE USED?
105262  001024                            BNE    PPP10            ;BRANCH IF NOT.
105264  022710  177777                    CMP    #-1,(R0)
105270  001021                            BNE    PPP10
105272  000421                            BR     PPPDONE ;GO TO NEXT TEST.

105274  177777  177777  177777    PPPBF0: .WORD  -1,-1,-1,-1,-1,-1
105302  177777  177777  177777

105310  177777  177777  177777    PPPBF1: .WORD  -1,-1,-1,-1,-1,-1
105316  177777  177777  177777

105324  123456  023456            PPPTP1: .WORD  123456,23456
105330  034567  045671                    .WORD  34567,45671

105334                            PPP10:
105334  104000                            EMT                     ;
105336                            PPPDONE:
105336  004767  017230                    JSR    PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
                                                                  ;SEE IF THE USER HAS EXPRESSED
                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                  ;THE USER TYPED CONTROL G?).





                                  ;*****************************************************************
                                  ;TEST 514          FDST MODE 2 TEST
                                  ;*****************************************************************
105342                            TS514:

                                  ;FIRST TEST STF.

105342  012700  177777                    MOV    #-1,R0  ;SET UP THE OUTPUT BUFFER.
105346  012701  105456                    MOV    #QQQBF0,R1
105352  012702  000014                    MOV    #14,R2
105356  010021                    QQQ2:   MOV    R0,(R1)+
105360  077202                            SOB    R2,QQQ2
```

```
105362  012700  000200                  MOV     #200,R0         ;SET FD MODE.
105366  170100                          LDFPS   R0
105370  012700  105506                  MOV     #QQQTP1,R0      ;SETUP AC0.
105374  172410                          LDD     (R0),AC0

105376  012700  105472                  MOV     #QQQBF1,R0      ;FDST ADDRESS.
105402  005002                          CLR     R2
105404  170102                          LDFPS   R2              ;SET FPS.
105406  174020                  QQQ3:   STF     AC0,(R0)+       ;TEST INSTRUCTION.

105410  022700  105476                  CMP     #QQQBF1+4,R0    ;WAS R0 INCREMENTED BY 4 PROPERLY?

105414  001401                          BEQ     QQQ4
105416  104000                          EMT                     ;REPORT R0 INCORRECT AFTER FDST MODE 2
105420  012700  105472          QQQ4:   MOV     #QQQBF1,R0      ;WAS THE OUTPUT DATA CORRECT?
105424  012701  105506                  MOV     #QQQTP1,R1
105430  022021                          CMP     (R0)+,(R1)+
105432  001031                          BNE     QQQ10           ;BRANCH IF INCORRECT.
105434  022021                          CMP     (R0)+,(R1)+
105436  001027                          BNE     QQQ10           ;BRANCH IF INCORRECT.
105440  022027  177777                  CMP     (R0)+,#-1       ;SEE IF ANY OTHER DATA BUFFER WORDS WERE MODIFIED.
105444  001024                          BNE     QQQ10           ;BRANCH IF INCORRECT.
105446  022027  177777                  CMP     (R0)+,#-1
105452  001021                          BNE     QQQ10           ;BRANCH IF INCORRECT.
105454  000421                          BR      QQQ20
105456  177777  177777  177777  QQQBF0: .WORD   -1,-1,-1,-1,-1,-1
105464  177777  177777  177777
105472  177777  177777  177777  QQQBF1: .WORD   -1,-1,-1,-1,-1,-1
105500  177777  177777  177777
105506  076543                  QQQTP1: 76543
105510  065432                          65432
105512  054321                          54321
105514  043210                          43210
                                ;REPORT OUTPUT DATA INCORRECT:
105516                          QQQ10:
105516  104000                          EMT                     ;


                                ;NOW TEST STD MODE 2.

105520  012700  105456          QQQ20:  MOV     #QQQBF0,R0      ;SET UP DEFAULT INPUT DATA BUFFER.
105524  010001                          MOV     R0,R1
105526  012702  000014                  MOV     #14,R2
105532  010021                  QQQ22:  MOV     R0,(R1)+
105534  077202                          SOB     R2,QQQ22
105536  012700  000200                  MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
105542  170100                          LDFPS   R0
105544  012700  105506                  MOV     #QQQTP1,R0      ;LOAD AC0.
105550  172410                          LDD     (R0),AC0
105552  012700  105472                  MOV     #QQQBF1,R0      ;SET DESTINATION ADDRESS.
105556  012737  105564  037364          MOV     #QQQ23,@#$TMP2
105564  174020                  QQQ23:  STD     AC0,(R0)+       ;TEST INSTRUCTION.
105566  022700  105502                  CMP     #QQQBF1+10,R0   ;WAS R0 INCREMENTED BY 10 CORRECTLY?
105572  001401                          BEQ     QQQ24
105574  104000                          EMT                     ;REPORT R0 INCORRECTLY INCREMENTED
105576  012700  105472          QQQ24:  MOV     #QQQBF1,R0      ;DID THE DATA REACH THE OUTPUT BUFFER CORRECTLY?
105602  012701  105506                  MOV     #QQQTP1,R1
```

```
105606 012702 000004           MOV     #4,R2
105612 022021           1$:    CMP     (R0)+,(R1)+
105614 001002                  BNE     QQQ25           ;BRANCH IF INCORRECT.
105616 077203                  SOB     R2,1$
105620 000401                  BR      QQQDONE
                        ;REPORT DATA INCORRECT.
105622                  QQQ25:
105622 104000                  EMT                     ;
105624                  QQQDONE:
105624 004767 016742           JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                       ;SEE IF THE USER HAS EXPRESSED
                                                       ;THE DESIRE TO CHANGE THE SOFTWARE
                                                       ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                       ;THE USER TYPED CONTROL G?).

                        ;*******************************************************************************
                        ;TEST 515        FDST MODE 2, WITH GR7, TEST
                        ;*******************************************************************************
105630                  TS515:

105630 012700 105700           MOV     #RRR3,R0        ;SET UP THE DATA BUFFER FOLLOWING THE TEST INSTRUCTION.
105634 012701 105746           MOV     #RRRTP1,R1
105640 012702 000004           MOV     #4,R2
105644 012021           1$:    MOV     (R0)+,(R1)+
105646 077202                  SOB     R2,1$
105650 012700 000200           MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
105654 170100                  LDFPS   R0
105656 012700 105756           MOV     #RRRTP2,R0      ;SET UP AC0.
105662 172410                  LDD     (R0),AC0
105664 012737 105744 000004    MOV     #RRR10,@#ERRVECT ;SET UP FOR AN ODD ADDRESS.
105672 005001                  CLR     R1
105674 005004                  CLR     R4
                        ;THIS IS THE TEST INSTRUCTION. IT SHOULD MODIFY THE FIRST LOCATION
                        ;AFTER IT TO BE AN INCREMENT R4, INC R4, INSTRUCTION INSTEAD
                        ;OF AN INCREMENT R1 INSTRUCTION. THE INCREMENT R4 SHOULD NOT BE
                        ;EXECUTED SINCE THE PC SHOULD BE INCREMENTED BY TWO DURING IMMEDIATE
                        ;MODE ADDRESSING. THUS AFTER THE EXECUTION OF THE NEXT 5 INSTRUCTIONS
                        ;R1 SHOULD CONTAIN 3 AND R4 SHOULD CONTAIN 0.
105676 174027           RRR2:  STD     AC0,(R7)+       ;TEST INSTRUCTION.
105700 005201           RRR3:  INC     R1              ;THE STD INSTRUCTION SHOULD CHANGE THIS TO INC R4.
105702 005201                  INC     R1
105704 005201                  INC     R1
105706 005201                  INC     R1
105710 012700 105766           MOV     #RRREXP,R0      ;SEE IF THE DATA WAS OUTPUT CORRECTLY.
105714 012702 105700           MOV     #RRR3,R2
105720 012703 000004           MOV     #4,R3
105724 022022           RRR4:  CMP     (R0)+,(R2)+
105726 001006                  BNE     RRR10           ;BRANCH IF INCORRECT.
105730 077303                  SOB     R3,RRR4
105732 005704                  TST     R4              ;MAKE SURE R4 IS 0.
105734 001003                  BNE     RRR10           ;BRANCH IF R4 IS INCORRECT.
105736 022701 000003           CMP     #3,R1           ;SEE IF R1 IS CORRECT.
105742 001415                  BEQ     RRRDONE
105744                  RRR10:
105744 104000                  EMT                     ;
                        ;THESE ARE TEST DATA PATTERNS USED TO SET UP THE OUTPUT BUFFER AT RRR3.
```

```
105746  005201                  RRRTP1: INC     R1
105750  005201                          INC     R1
105752  005201                          INC     R1
105754  005201                          INC     R1
                                ;THIS IS THE DATA PUT IN AC0 BEFORE EXECUTION OF THE STD.
105756  005204                  RRRTP2: INC     R4
105760  005204                          INC     R4
105762  005204                          INC     R4
105764  005204                          INC     R4
                                ;THIS IS THE EXPECTED DATA AT RRR3 AFTER EXECUTION OF THE STD.
105766  005204                  RRREXP: INC     R4
105770  005201                          INC     R1
105772  005201                          INC     R1
105774  005201                          INC     R1
105776                          RRRDONE:
105776  004767  016570                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*************************************************************************
                                ;TEST 516       FDST MODE 4 TEST
                                ;*************************************************************************
106002                          TS516:

106002  012700  177777                  MOV     #-1,R0          ;SET UP THE OUTPUT BUFFER.
106006  012701  106130                  MOV     #SSSBF0,R1
106012  012702  000010                  MOV     #10,R2
106016  010021                  1$:     MOV     R0,(R1)+
106020  077202                          SOB     R2,1$
106022  012700  000200                  MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
106026  170100                          LDFPS   R0
106030  012700  106150                  MOV     #SSSTP1,R0      ;SET UP AC0.
106034  172410                          LDD     (R0),AC0
106036  012737  106160  000004          MOV     #SSS10,@#ERRVECT ;SET UP FOR A TRAP TO 4.
106044  012700  106140                  MOV     #SSSA1,R0       ;SET UP THE DESTINATION ADDRESS.

106050  174040                  SSS2:   STD     AC0,-(R0)       ;TEST INSTRUCTION.
106052  005201                          INC     R1
106054  020027  106130                  CMP     R0,#SSSBF0      ;SEE IF R0 WAS DECREMENTED PROPERLY.
106060  001037                          BNE     SSS10           ;BRANCH IF R0 IS INCORRECT.
106062  012700  106130                  MOV     #SSSBF0,R0      ;WAS THE OUTPUT DATA CORRECT?
106066  012701  106150                  MOV     #SSSTP1,R1
106072  012702  000004                  MOV     #4,R2
106076  022021                  1$:     CMP     (R0)+,(R1)+
106100  001027                          BNE     SSS10           ;BRANCH IF INCORRECT.
106102  077203                          SOB     R2,1$
106104  012700  177777                  MOV     #-1,R0          ;IS THE REST OF THE OUTPUT BUFFER CORRECT, -1?
106110  012701  106140                  MOV     #SSSA1,R1
106114  012702  000004                  MOV     #4,R2
106120  020021                  2$:     CMP     R0,(R1)+
106122  001016                          BNE     SSS10           ;BRANCH IF INCORRECT.
106124  077203                          SOB     R2,2$
106126  000415                          BR      SSSDONE
```

```
                                        ;THIS IS THE OUTPUT DATA BUFFER.
106130  177777                          SSSBF0: -1
106132  177777                                  -1
106134  177777                                  -1
106136  177777                                  -1
106140  177777                          SSSA1:  -1
106142  177777                                  -1
106144  177777                                  -1
106146  177777                                  -1

                                        ;THIS IS THE TEST DATA LOADED INTO AC0:
106150  147250                          SSSTP1: 147250
106152  036147                                  36147
106154  025036                                  25036
106156  147250                                  147250

106160                                  SSS10:
106160  104000                                  EMT                             ;
106162                                  SSSDONE:
106162  004767  016404                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).

                                        ;*******************************************************************************
                                        ;TEST 517       FDST MODE 3 TEST
                                        ;*******************************************************************************
106166                                  TS517:

106166  012701  106276                          MOV     #TTTBF0,R1              ;SET UP THE OUTPUT DATA BUFFER.
106172  012700  177777                          MOV     #-1,R0
106176  012702  000013                          MOV     #13,R2
106202  010021                          1$:     MOV     R0,(R1)+
106204  077202                                  SOB     R2,1$
106206  012737  106276  106312                  MOV     #TTTBF0,@#TTTA2
106214  012700  000200                          MOV     #200,R0                 ;ENTER DOUBLE FLOATING MODE.
106220  170100                                  LDFPS   R0
106222  012700  106314                          MOV     #TTTTP1,R0              ;SET UP AC0.
106226  172410                                  LDD     (R0),AC0
106230  012737  106324  000004                  MOV     #TTT10,@#ERRVECT        ;SET UP FOR TRAPS TO 4.
106236  012700  106312                          MOV     #TTTA2,R0               ;SET UP THE DESTINATION ADDRESS.

106242  174030                          TTT2:   STD     AC0,@(R0)+              ;TEST INSTRUCTION.

106244  020027  106314                          CMP     R0,#TTTA2+2             ;SEE IF R0 WAS INCREMENTED CORRECTLY.
106250  001025                                  BNE     TTT10                   ;BRANCH IF INCORRECT.
106252  012701  106276                          MOV     #TTTBF0,R1              ;CHECK THE OUTPUT DATA BUFFER.
106256  012702  106314                          MOV     #TTTTP1,R2
106262  012703  000004                          MOV     #4,R3
106266  022122                          TTT3:   CMP     (R1)+,(R2)+
106270  001015                                  BNE     TTT10                   ;BRANCH IF NOT CORRECT.
106272  077303                                  SOB     R3,TTT3
106274  000414                                  BR      TTTDONE

                                        ;THIS IS THHE OUTPUT DATA BUFFER:
```

```
106276  177777                          TTTBF0: -1
106300  177777                                  -1
106302  177777                                  -1
106304  177777                                  -1
106306  177777                                  -1
106310  177777                          TTTA1:  -1
106312  106276                          TTTA2:  TTTBF0
106314  101213                          TTTTP1: 101213
106316  141516                                  141516
106320  071727                                  71727
106322  037475                                  37475

106324                                  TTT10:
106324  104000                                  EMT                             ;

106326                                  TTTDONE:
106326  004767  016240                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).

                                        ;*******************************************************************************
                                        ;TEST 520       FDST MODE 5 TEST
                                        ;*******************************************************************************
106332                                  TS520:

106332  012701  106442                          MOV     #UUUBF0,R1              ;SET UP THE OUTPUT DATA BUFFER.
106336  012700  177777                          MOV     #-1,R0
106342  012702  000013                          MOV     #13,R2
106346  010021                          1$:     MOV     R0,(R1)+
106350  077202                                  SOB     R2,1$
106352  012737  106442  106454                  MOV     #UUUBF0,@#UUUA1
106360  012700  000200                          MOV     #200,R0                 ;ENTER DOUBLE FLOATING MODE.
106364  170100                                  LDFPS   R0
106366  012700  106460                          MOV     #UUUTP1,R0              ;SET UP AC0.
106372  172410                                  LDD     (R0),AC0
106374  012737  106470  000004                  MOV     #UUU10,@#ERRVECT        ;GET READY FOR ANY TRAPS TO 4.
106402  012700  106456                          MOV     #UUUA2,R0               ;SET UP THE DESTINATION ADDRESS.
106406  174050                          UUU2:   STD     AC0,@-(R0)              ;TEST INSTRUCTION.
106410  020027  106454                          CMP     R0,#UUUA2-2             ;WAS R0 DECRIMENTED PROPERLY?
106414  001025                                  BNE     UUU10                   ;BRANCH IF R0 IS INCORRECT.
106416  012701  106442                          MOV     #UUUBF0,R1              ;WAS THE DATA OUTPUT CORRECTLY?
106422  012702  106460                          MOV     #UUUTP1,R2
106426  012703  000004                          MOV     #4,R3
106432  022122                          UUU3:   CMP     (R1)+,(R2)+
106434  001015                                  BNE     UUU10                   ;BRANCH IF DATA IS INCORRECT.
106436  077303                                  SOB     R3,UUU3
106440  000414                                  BR      UUUDONE

                                        ;THIS IS THE OUTPUT DATA BUFFER
106442  177777                          UUUBF0: -1
106444  177777                                  -1
106446  177777                                  -1
106450  177777                                  -1
106452  177777                                  -1
```

```
106454  106442                          UUUA1:  UUUBF0
106456  177777                          UUUA2:  -1
106460  020212                          UUUTP1: 20212
106462  023242                                  23242
106464  026273                                  26273
106466  031323                                  031323

106470                                  UUU10:
106470  104000                                  EMT                             ;
106472                                  UUUDONE:
106472  004767  016074                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).

                                        ;*************************************************************************
                                        ;TEST 521       FDST MODE 6, INDEX MODE, TEST
                                        ;*************************************************************************
106476                                  TS521:

106476  012700  000200                          MOV     #200,R0                 ;ENTER DOUBLE FLOATING MODE.
106502  170100                                  LDFPS   R0
106504  012701  106606                          MOV     #VVVBF0,R1              ;SET UP THE OUT PUT DATA BUFFER.
106510  012700  177777                          MOV     #-1,R0
106514  012702  000004                          MOV     #4,R2
106520  010021                          1$:     MOV     R0,(R1)+
106522  077202                                  SOB     R2,1$
106524  012737  106626  000004                  MOV     #VVV10,@#ERRVECT        ;SET UP VECTOR 4 INCASE OF ERROR.
106532  012700  106616                          MOV     #VVVTP1,R0              ;SET UP AC0.
106536  172410                                  LDD     (R0),AC0
106540  012700  100705                          MOV     #VVVBF0-5701,R0         ;SET UP THE DESTINATION ADDRESS.
106544  012701  000001                          MOV     #1,R1
106550  174060  005701                  VVV2:   STD     AC0,5701(R0)            ;TEST INSTRUCTION.

106554  020027  100705                          CMP     R0,#VVVBF0-5701         ;SEE IF R0 WAS MODIFIED.
106560  001022                                  BNE     VVV10                   ;BRANCH IF INCORRECT.
106562  012702  106606                          MOV     #VVVBF0,R2              ;WAS THE OUTPUT DATA CORRECT.
106566  012703  106616                          MOV     #VVVTP1,R3
106572  012704  000004                          MOV     #4,R4
106576  022223                          1$:     CMP     (R2)+,(R3)+
106600  001012                                  BNE     VVV10                   ;BRANCH IF INCORRECT DATA.
106602  077403                                  SOB     R4,1$
106604  000411                                  BR      VVVDONE
106606  177777                          VVVBF0: -1
106610  177777                                  -1
106612  177777                                  -1
106614  177777                                  -1
106616  030313                          VVVTP1: 30313
106620  023334                                  23334
106622  035363                                  35363
106624  074041                                  74041

106626                                  VVV10:
106626  104000                                  EMT                             ;
106630                                  VVVDONE:
```

```
106630  004767  015736                    JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                  ;SEE IF THE USER HAS EXPRESSED
                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                  ;THE USER TYPED CONTROL G?).

                                  ;*******************************************************************************
                                  ;TEST 522         FDST MODE 7, INDEX DEFERRED MODE, TEST
                                  ;*******************************************************************************
106634                            TS522:

106634  012700  000200                    MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
106640  170100                            LDFPS   R0
106642  012701  106752                    MOV     #WWWBF0,R1      ;SET UP THE OUTPUT DATA BUFFER.
106646  012700  177777                    MOV     #-1,R0
106652  012702  000004                    MOV     #4,R2
106656  010021                    1$:     MOV     R0,(R1)+
106660  077202                            SOB     R2,1$
106662  012737  107002  000004            MOV     #WWW10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
106670  012700  106762                    MOV     #WWWTP1,R0      ;SET UP AC0.
106674  172410                            LDD     (R0),AC0
106676  012700  101071                    MOV     #WWWBF1-5701,R0 ;SET UP THE DESTINATION ADDRESS.
106702  012701  000001                    MOV     #1,R1
106706  012737  106752  106772            MOV     #WWWBF0,@#WWWBF1
106714  174070  005701            WWW2:   STD     AC0,@5701(R0)   ;TEST INSTRUCTION.

106720  020027  101071                    CMP     R0,#WWWBF1-5701 ;IS R0 CORRECT?
106724  001026                            BNE     WWW10           ;BRANCH IF INCORRECT.
106726  012702  106752                    MOV     #WWWBF0,R2      ;WAS THE DATA OUTPUT CORRECTLY?
106732  012703  106762                    MOV     #WWWTP1,R3
106736  012704  000004                    MOV     #4,R4
106742  022223                    1$:     CMP     (R2)+,(R3)+
106744  001016                            BNE     WWW10           ;BRANCH IF DATA IS INCORRECT.
106746  077403                            SOB     R4,1$
106750  000415                            BR      WWWDONE
106752  177777                    WWWBF0: -1
106754  177777                            -1
106756  177777                            -1
106760  177777                            -1
106762  041424                    WWWTP1: 41424
106764  034445                            34445
106766  046475                            46475
106770  051525                            051525
106772  177777                    WWWBF1: -1
106774  177777                            -1
106776  177777                            -1
107000  177777                            -1

107002                            WWW10:
107002  104000                            EMT                     ;
107004                            WWWDONE:
107004  004767  015562                    JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                  ;SEE IF THE USER HAS EXPRESSED
                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                  ;THE USER TYPED CONTROL G?).
```

```

                                          ;*******************************************************************
                                          ;TEST 523          STCFD TEST
                                          ;*******************************************************************
107010                                    TS523:

                                          ;AC=0
107010  004767  000262                    XXX1:   JSR     PC,STCFDS
107014  000000                            1$:     0                ;AC
107016  000000                                    0
107020  000000                                    0
107022  000000                                    0
107024  000000                            2$:     0                ;RES
107026  000000                                    0
107030  000000                                    0
107032  000000                                    0
107034  000000                            3$:     0                ;ERROR RES.
107036  000000                                    0
107040  177777                                    -1
107042  177777                                    -1
107044  047000                            4$:     47000            ;FPS BEFORE EXECUTION.
107046  047004                                    47004            ;FPS AFTER EXECUTION.
107050  177777                                    -1               ;FEC
107052  147004                                    147004           ;ERROR FPS.


107054  004767  000216                    XXX2:   JSR     PC,STCFDS
107060  017203                            1$:     17203            ;AC
107062  142536                                    142536
107064  047506                                    47506
107066  172031                                    172031
107070  017203                            2$:     17203            ;RES
107072  142536                                    142536
107074  000000                                    0
107076  000000                                    0
107100  017203                            3$:     17203            ;ERROR RES.
107102  142536                                    142536
107104  047506                                    47506
107106  172031                                    172031
107110  040000                            4$:     40000            ;FPS BEFORE EXECUTION.
107112  040000                                    40000            ;FPS AFTER EXECUTION.
107114  177777                                    -1               ;FEC
107116  177777                                    -1               ;ERROR FPS.


107120  004767  000152                    XXX3:   JSR     PC,STCFDS
107124  050717                            1$:     50717            ;AC
107126  027374                                    27374
107130  075767                                    75767
107132  077071                                    77071
107134  050717                            2$:     50717            ;RES
107136  027374                                    27374
107140  000000                                    0
107142  000000                                    0
107144  000000                            3$:     0                ;ERROR RES.
107146  000000                                    0
```

```
107150  000000                           0
107152  000000                           0
107154  047000                  4$:      47000           ;FPS BEFORE EXECUTION.
107156  047000                           47000           ;FPS AFTER EXECUTION.
107160  177777                           -1              ;FEC
107162  174002                           174002          ;ERROR FPS.


107164  004767  000106          XXX4:    JSR     PC,STCFDS
107170  020212                  1$:      20212           ;AC
107172  032425                           32425
107174  026272                           26272
107176  002123                           02123
107200  020212                  2$:      20212           ;RES
107202  032425                           32425
107204  000000                           0
107206  000000                           0
107210  020212                  3$:      20212           ;ERROR RES.
107212  032425                           32425
107214  100000                           100000
107216  000000                           0
107220  040000                  4$:      40000           ;FPS BEFORE EXECUTION.
107222  040000                           40000           ;FPS AFTER EXECUTION.
107224  177777                           -1              ;FEC
107226  177777                           -1              ;ERROR FPS.


107230  004767  000042          XXX5:    JSR     PC,STCFDS
107234  121314                  1$:      121314          ;AC
107236  151617                           151617
107240  101112                           101112
107242  131415                           131415
107244  121314                  2$:      121314          ;RES
107246  151617                           151617
107250  000000                           0
107252  000000                           0
107254  021314                  3$:      21314           ;ERROR RES.
107256  151617                           151617
107260  000000                           0
107262  000000                           0
107264  040000                  4$:      40000           ;FPS BEFORE EXECUTION.
107266  040010                           40010           ;FPS AFTER EXECUTION.
107270  177777                           -1              ;FEC
107272  177777                           -1              ;ERROR FPS.
107274  000460                  6$:      BR      XXXDONE


                                ;


                                ;THIS SUBROUTINE, STCFDS, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE STCFD INSTRUCTION AND CHECK THE RESULTS. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;                JSR     PC,@#STCFDS
                                ;                ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                ;                RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
```

```
                            ;                       ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
                            ;                       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                            ;                       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                            ;                       FEC:    .WORD   X               ;EXPECTED FEC
                            ;                       ERFPS:  .WORD   X               ;ERROR FPS.
                            ;                       ERR1:   ERROR   X               ;DATA ERROR.
                            ;                               BR      CONT
                            ;                       ERR2:   ERROR   X               ;FPS ERROR.
                            ;                       CONT:                           ;RETURN ADDRESS
                            ;
                            ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                            ;THE STCFD INSTRUCTION IS EXECUTED.
                            ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                            ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
                            ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
                            ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
                            ;TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
                            ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                            ;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                            ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                            ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
                            ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                            ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
                            ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

107276  012601              STCFDS: MOV     (SP)+,R1                ;PICK UP THE POINTER TO THE OPERANDS.
107300  012700  000200              MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
107304  170100                      LDFPS   R0
107306  010100                      MOV     R1,R0           ;LOAD AC0.
107310  172410                      LDD     (R0),AC0
107312  012700  177777              MOV     #-1,R0          ;FILL THE OUTPUT BUFFER WITH -1'S.
107316  012702  107426              MOV     #STCFT,R2
107322  012703  000004              MOV     #4,R3
107326  010022              1$:     MOV     R0,(R2)+
107330  077302                      SOB     R3,1$
107332  016100  000030              MOV     30(R1),R0       ;LOAD THE FPS.
107336  170100                      LDFPS   R0
107340  012700  107426              MOV     #STCFT,R0       ;SET UP THE DESTINATION ADDRESS.
107344  176010              2$:     STCFD   AC0,(R0)        ;TEST INSTRUCTION.

107346  170204                      STFPS   R4              ;GET THE FPS.
107350  170305                      STST    R5              ;GET THE FEC.
107352  010102                      MOV     R1,R2           ;CHECK THE RESULT.
107354  062702  000010              ADD     #10,R2
107360  012703  107426              MOV     #STCFT,R3
107364  012700  000004              MOV     #4,R0
107370  022223              3$:     CMP     (R2)+,(R3)+
107372  001014                      BNE     10$             ;BRANCH IF INCORRECT.
107374  077003                      SOB     R0,3$

107376  016102  000032              MOV     32(R1),R2
107402  020204                      CMP     R2,R4           ;IS THE FPS CORRECT?
107404  001007                      BNE     10$             ;BRANCH IF FPS INCORRECT.
107406  005702                      TST     R2              ;IF EXPECTED FPS IS NEGATIVE, THEN
107410  100003                      BPL     4$              ;GO AHEAD AND CHECK THE FEC.
107412  026105  000036              CMP     36(R1),R5
```

```
107416  001002                          BNE     10$             ;BRANCH IF FEC IS INCORRECT.
107420  000161  000040          4$:     JMP     40(R1)          ;RETURN.
107424                          10$:
107424  104000                          EMT                     ;
107426  177777  177777  177777  STCFT:  -1,-1,-1,-1
107434  177777
107436                          XXXDONE:
107436  004767  015130                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;************************************************************************
                                ;TEST 524        STCDF TEST
                                ;************************************************************************
107442                          TS524:

                                ;AC=0
107442  004767  000262          YYY1:   JSR     PC,STCDFS
107446  000000                  1$:     0                       ;AC
107450  000000                          0
107452  000000                          0
107454  000000                          0
107456  000000                  2$:     0                       ;RES
107460  000000                          0
107462  177777                          -1
107464  177777                          -1
107466  000000                  3$:     0                       ;ERROR RES.
107470  000000                          0
107472  000000                          0
107474  000000                          0
107476  047200                  4$:     47200                   ;FPS BEFORE EXECUTION.
107500  047204                          47204                   ;FPS AFTER EXECUTION.
107502  177777                          -1                      ;FEC
107504  177777                          -1                      ;ERROR FPS.


107506  004767  000216          YYY2:   JSR     PC,STCDFS
107512  067574                  1$:     67574                   ;AC0
107514  073727                          73727
107516  170777                          170777
107520  067574                          67574
107522  067574                  2$:     67574                   ;RES
107524  073730                          73730
107526  177777                          -1
107530  177777                          -1
107532  067574                  3$:     67574                   ;ERROR RES.
107534  073727                          73727
107536  177777                          -1
107540  177777                          -1
107542  040200                  4$:     40200                   ;FPS BEFORE EXECUTION.
107544  040200                          40200                   ;FPS AFTER EXECUTION.
107546  177777                          -1                      ;FEC
107550  177777                          -1                      ;ERROR FPS.
```

```

107552 004767 000152    YYY3:   JSR     PC,STCDFS
107556 077777           1$:     77777                   ;AC0
107560 177777                   -1
107562 100000                   100000
107564 000000                   0
107566 000000           2$:     0                       ;RES
107570 000000                   0
107572 177777                   -1
107574 177777                   -1
107576 077777           3$:     77777                   ;ERROR RES.
107600 177777                   -1
107602 177777                   -1
107604 177777                   -1
107606 040200           4$:     40200                   ;FPS BEFORE EXECUTION.
107610 040206                   40206                   ;FPS AFTER EXECUTION.
107612 177777                   -1                      ;FEC
107614 040204                   40204                   ;ERROR FPS.


107616 004767 000106    YYY4:   JSR     PC,STCDFS
107622 077777           1$:     77777                   ;AC0
107624 177777                   -1
107626 100000                   100000
107630 000000                   0
107632 000000           2$:     0                       ;RES
107634 000000                   0
107636 177777                   -1
107640 177777                   -1
107642 077777           3$:     77777                   ;ERROR RES.
107644 177777                   -1
107646 177777                   -1
107650 177777                   -1
107652 040200           4$:     40200                   ;FPS BEFORE EXECUTION.
107654 040206                   40206                   ;FPS AFTER EXECUTION.
107656 177777                   -1                      ;FEC
107660 140206                   140206                  ;ERROR FPS.


107662 004767 000042    YYY5:   JSR     PC,STCDFS
107666 177777           1$:     177777                  ;AC0
107670 177777                   -1
107672 100000                   100000
107674 000000                   0
107676 100000           2$:     100000                  ;RES
107700 000000                   0
107702 177777                   -1
107704 177777                   -1
107706 000000           3$:     0                       ;ERROR RES.
107710 000000                   0
107712 177777                   -1
107714 177777                   -1
107716 047200           4$:     47200                   ;FPS BEFORE EXECUTION.
107720 147216                   147216                  ;FPS AFTER EXECUTION.
107722 000010                   10                      ;FEC
107724 047206                   47206                   ;ERROR FPS.
```

```
107726  000460                  6$:     BR      YYYDONE
                                ;THIS SUBROUTINE, STCDFS, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE STCDF INSTRUCTION AND CHECK THE RESULTS. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               JSR     PC,@#STCFDS
                                ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
                                ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               FEC:    .WORD   X               ;EXPECTED FEC
                                ;               ERFPS:  .WORD   X               ;ERROR FPS.
                                ;               ERR1:   ERROR   X               ;DATA ERROR.
                                ;                       BR      CONT
                                ;               ERR2:   ERROR   X               ;FPS ERROR.
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE STCFD INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
                                ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
                                ;TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

107730  012601                  STCDFS: MOV     (SP)+,R1        ;PICK UP THE POINTER TO THE OPERANDS.
107732  012700  000200                  MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
107736  170100                          LDFPS   R0
107740  010100                          MOV     R1,R0           ;LOAD AC0.
107742  172410                          LDD     (R0),AC0
107744  012700  177777                  MOV     #-1,R0          ;FILL THE OUTPUT BUFFER WITH -1'S.
107750  012702  110060                  MOV     #STCDT,R2
107754  012703  000004                  MOV     #4,R3
107760  010022                  1$:     MOV     R0,(R2)+
107762  077302                          SOB     R3,1$
107764  016100  000030                  MOV     30(R1),R0       ;LOAD THE FPS.
107770  170100                          LDFPS   R0
107772  012700  110060                  MOV     #STCDT,R0       ;SET UP THE DESTINATION ADDRESS.
107776  176010                  2$:     STCDF   AC0,(R0)        ;TEST INSTRUCTION.

110000  170204                          STFPS   R4              ;GET THE FPS.
110002  170305                          STST    R5              ;GET THE FEC.
110004  010102                          MOV     R1,R2           ;CHECK THE RESULT.
110006  062702  000010                  ADD     #10,R2
110012  012703  110060                  MOV     #STCDT,R3
110016  012700  000004                  MOV     #4,R0
110022  022223                  3$:     CMP     (R2)+,(R3)+
110024  001014                          BNE     10$             ;BRANCH IF INCORRECT.
```

```
110026 077003                      SOB     R0,3$

110030 016102 000032               MOV     32(R1),R2
110034 020204                      CMP     R2,R4           ;IS THE FPS CORRECT?
110036 001007                      BNE     10$             ;BRANCH IF FPS INCORRECT.
110040 005702                      TST     R2              ;IF EXPECTED FPS IS NEGATIVE, THEN
110042 100003                      BPL     4$              ;GO AHEAD AND CHECK THE FEC.
110044 026105 000034               CMP     34(R1),R5
110050 001002                      BNE     10$             ;BRANCH IF FEC IS INCORRECT.
110052 000161 000040       4$:     JMP     40(R1)          ;RETURN.
110056                     10$:
110056 104000                      EMT                     ;
110060 177777 177777 177777 STCDT: -1,-1,-1,-1
110066 177777
110070                     YYYDONE:
110070 004767 014476               JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                           ;SEE IF THE USER HAS EXPRESSED
                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                           ;THE USER TYPED CONTROL G?).
                           ;*********************************************************************
                           ;TEST 525          STCFD WITH ILLEGAL ACCUMULATOR TEST
                           ;*********************************************************************
110074                     TS525:

110074 012700 040000               MOV     #40000,R0       ;DISSABLE INTERRUPTS.
110100 170100                      LDFPS   R0
110102 176006              ZZZ2:   STCFD   AC0,AC6         ;THIS TEST INSTRUCTION SHOULD CAUSE AN ERROR.

110104 170204                      STFPS   R4              ;GET FPS.
110106 170305                      STST    R5              ;GET FEC.
110110 020427 140000               CMP     R4,#140000      ;IS FPS CORRECT?
110114 001004                      BNE     ZZZ10           ;BRANCH IF INCORRECT FPS.
110116 022705 000002               CMP     #2,R5           ;IS FEC CORRECT?
110122 001001                      BNE     ZZZ10           ;BRANCH IF INCORRECT.
110124 000401                      BR      ZZZDONE

110126                     ZZZ10:
110126 104000                      EMT                     ;

110130                     ZZZDONE:
110130 004767 014436               JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                           ;SEE IF THE USER HAS EXPRESSED
                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                           ;THE USER TYPED CONTROL G?).

                           ;*********************************************************************
                           ;TEST 526          CLRD TEST
                           ;*********************************************************************
110134                     TS526:
110134 012700 110240               MOV     #AABTP1,R0      ;SET UP OUTPUT BUFFER
110140 012701 110230               MOV     #AABBF0,R1
110144 012702 000004               MOV     #4,R2
110150 012021              1$:     MOV     (R0)+,(R1)+
110152 077202                      SOB     R2,1$
```

```
110154  012700  110230           MOV     #AABBF0,R0        ;SET UP DESTINATION OPERAND ADDRESS.
110160  012701  000213           MOV     #213,R1           ;SET UP FPS.
110164  170101                   LDFPS   R1
110166  170410          2$:      CLRD    (R0)              ;TEST INSTRUCTION.

110170  170205                   STFPS   R5                ;GET FPS.
110172  012702  000004           MOV     #4,R2             ;SEE IF RESULT CLEAR, 0.
110176  012701  110230           MOV     #AABBF0,R1
110202  005721          3$:      TST     (R1)+
110204  001010                   BNE     AAB2              ;BRANCH IF RESULT INCORRECT, NOT 0.
110206  077203                   SOB     R2,3$
110210  022705  000204           CMP     #204,R5           ;SEE IF FPS IS CORRECT.
110214  001004                   BNE     AAB2              ;BRANCH IF INCORRECT.
110216  020027  110230           CMP     R0,#AABBF0        ;SEE IF R0 IS CORRECT.
110222  001001                   BNE     AAB2              ;BRANCH IF R0 IS INCORRECT.
110224  000411                   BR      AABDONE


110226                  AAB2:
110226  104000                   EMT                       ;

                        ;THIS IS THE TEST DATA BUFFER, OUTPUT DATA BUFFER.
110230  073475          AABBF0:  73475
110232  067707                   67707
110234  127347                   127347
110236  056770                   56770
                        ;THIS IS THE DATA USED TO SET UP THE OUTPUT BUFFER.
110240  073475          AABTP1:  73475
110242  067707                   67707
110244  127347                   127347
110246  056770                   56770
110250                  AABDONE:
110250  004767  014316           JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                           ;SEE IF THE USER HAS EXPRESSED
                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                           ;THE USER TYPED CONTROL G?).

                        ;*************************************************************************
                        ;TEST 527        CLRD WITH ILLEGAL ACCUMULATOR TEST
                        ;*************************************************************************
110254                  TS527:
110254  012700  040200           MOV     #40200,R0         ;SET UP THE FPS, NO INTERRUPTS AND FD=1.
110260  170100                   LDFPS   R0
110262  170407          CCB2:    CLRD    AC7               ;TEST INSTRUCTION.

110264  170204                   STFPS   R4                ;GET FPS.
110266  170305                   STST    R5                ;GET FEC.
110270  020427  140200           CMP     R4,#140200              ;IS THE FPS CORRECT?
110274  001004                   BNE     CCB10             ;BRANCH IF FPS IS INCORRECT.
110276  022705  000002           CMP     #2,R5             ;IS THE FEC CORRECT?
110302  001001                   BNE     CCB10             ;BRANCH IF FEC IS INCORRECT.
110304  000401                   BR      CCBDONE

110306                  CCB10:
110306  104000                   EMT                       ;
```

```
110310                          CCBDONE:
110310  004767  014256                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 530        NEGF, ABSF AND TSTF SOURCE MODE 0 WITH ILLEGAL AC7, TEST
                                ;*******************************************************************************
110314                          TS530:

110314  012700  040200                  MOV     #40200,R0               ;SET UP THE FPS, FID=1 AND FD=1.
110320  170100                          LDFPS   R0
110322  170707                  VVB2:   NEGD    AC7             ;TEST INSTRUCTION.

110324  170204                          STFPS   R4              ;GET FPS.
110326  170305                          STST    R5              ;GET FEC.

110330  022704  140200                  CMP     #140200,R4              ;IS FPS CORRECT?
110334  001004                          BNE     VVB10           ;BRANCH IF FPS IS INCORRECT.
110336  022705  000002                  CMP     #2,R5           ;IS FEC CORRECT?
110342  001001                          BNE     VVB10           ;BRANCH IF FEC IS INCORRECT.
110344  000401                          BR      VVBDONE
110346                          VVB10:
110346  104000                          EMT                     ;

110350                          VVBDONE:
110350  004767  014216                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 531        NEGF, ABSF AND TSTF SOURCE MODE 0 TEST
                                ;*******************************************************************************
110354                          TS531:

110354  012700  000200                  MOV     #200,R0         ;SET FD MODE.
110360  170100                          LDFPS   R0
110362  012700  110454                  MOV     #DDBTP1,R0      ;SET UP AC0.
110366  172410                          LDD     (R0),AC0                ;SET AC0 = 0
110370  005000                          CLR     R0              ;CLEAR THE FPS.
110372  170100                          LDFPS   R0
110374  012700  110464                  MOV     #DDBTP2,R0              ;LOAD AC0 TO BE A FLOATING 0.
110400  172410                          LDF     (R0),AC0        ;SET AC0=ZERO
                                                                ;FLOAT
110402  012700  000201                  MOV     #201,R0         ;SET FD MODE.
110406  170100                          LDFPS   R0
110410  170700                  DDB2:   NEGD    AC0             ;TEST INSTRUCTION

110412  170205                          STFPS   R5              ;GET FPS.
110414  012700  000200                  MOV     #200,R0         ;SET FD MODE.
110420  170100                          LDFPS   R0
110422  012700  110474                  MOV     #DDBBF0,R0      ;GET THE RESULT OUT OF AC0.
```

```
110426  174010                        STD     AC0,(R0)
                                                                ;SEE IF THE RESULT IS CORRECT.
110430  012701  000004                MOV     #4,R1
110434  005720                1$:     TST     (R0)+
110436  001005                        BNE     DDB5              ;BRANCH IF THE RESULT IS INCORRECT.
110440  077103                        SOB     R1,1$
110442  022705  000204                CMP     #204,R5           ;IS THE FPS CORRECT?
110446  001001                        BNE     DDB5              ;BRANCH IF THE FPS IS INCORRECT.
110450  000415                        BR      DDBDONE
110452                        DDB5:
110452  104000                        EMT                       ;

                              ;THESE ARE TEST DATA TABLES AND AN OUTPUT BUFFER.
110454  101112                DDBTP1: 101112
110456  131415                        131415
110460  161710                        161710
110462  111213                        111213
110464  000000                DDBTP2: 0
110466  000000                        0
110470  000000                        0
110472  000000                        0

110474  177777                DDBBF0: -1
110476  177777                        -1
110500  177777                        -1
110502  177777                        -1

110504                        DDBDONE:
110504  004767  014062                JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                              ;*************************************************************************
                              ;TEST 532        NEGF, ABSF AND TSTF SOURCE MODE 1 TEST
                              ;*************************************************************************
110510                        TS532:

110510  012700  110610                MOV     #EEBTP1,R0        ;SET UP THE DATA BUFFER.
110514  012701  110630                MOV     #EEBBF1,R1
110520  012702  000004                MOV     #4,R2
110524  012021                1$:     MOV     (R0)+,(R1)+
110526  077202                        SOB     R2,1$
110530  012700  000200                MOV     #200,R0           ;SET FD MODE.
110534  170100                        LDFPS   R0
110536  012700  110630                MOV     #EEBBF1,R0        ;SET UP THE OPERAND ADDRESS.
110542  012737  110640  000004        MOV     #EEB10,@#ERRVECT  ;SET UP VECTOR 4 IN CASE OF ERROR.
110550  170710                EEB2:   NEGD    (R0)              ;TEST INSTRUCTION.

110552  170205                        STFPS   R5                ;GET FPS.
110554  012701  110630                MOV     #EEBBF1,R1        ;SEE IF RESULT IS CORRECT.
110560  012702  000004                MOV     #4,R2
110564  005721                1$:     TST     (R1)+
110566  001024                        BNE     EEB10             ;BRANCH IF NOT CORRECT.
110570  077203                        SOB     R2,1$
```

```
110572  020027  110630            CMP     R0,#EEBBF1        ;IS R0 CORRECT?
110576  001020                    BNE     EEB10                     ;BRANCH IF NOT CORRECT.
110600  022705  000204            CMP     #204,R5           ;IS THE FPS CORRECT?
110604  001015                    BNE     EEB10             ;BRANCH IF NOT CORRECT.
110606  000415                    BR      EEBDONE

                          ;THESE ARE TEST DATA TABLES AND A BUFFER.
110610  000177            EEBTP1: 177
110612  167574                    167574
110614  137271                    137271
110616  107675                    107675
110620  177777            EEBBF0: -1
110622  177777                    -1
110624  177777                    -1
110626  177777                    -1
110630  177777            EEBBF1: -1
110632  177777                    -1
110634  177777                    -1
110636  177777                    -1
110640                    EEB10:
110640  104000                    EMT                       ;
110642                    EEBDONE:
110642  004767  013724            JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                            ;SEE IF THE USER HAS EXPRESSED
                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                            ;THE USER TYPED CONTROL G?).

                          ;*************************************************************************
                          ;TEST 533         NEGF, ABSF AND TSTF SOURCE MODE 2 TEST
                          ;*************************************************************************
110646                    TS533:

110646  012700  110746            MOV     #FFBTP1,R0                ;SET UP THE DATA BUFFER.
110652  012701  110756            MOV     #FFBBF1,R1
110656  012702  000004            MOV     #4,R2
110662  012021            1$:     MOV     (R0)+,(R1)+
110664  077202                    SOB     R2,1$
110666  012700  000200            MOV     #200,R0           ;SET FD.
110672  170100                    LDFPS   R0
110674  012700  110756            MOV     #FFBBF1,R0        ;SET UP THE OPERAND ADDRESS.
110700  012737  110766  000004    MOV     #FFB10,@#ERRVECT  ;SET UP VECTOR 4 IN CASE OF AN ERROR.

110706  170620            FFB2:   ABSD    (R0)+             ;TEST INSTRUCTION.

110710  170205                    STFPS   R5                ;GET FPS.
110712  012701  110756            MOV     #FFBBF1,R1                ;CHECK RESULT.
110716  012702  000004            MOV     #4,R2
110722  005721            1$:     TST     (R1)+
110724  001020                    BNE     FFB10             ;BRANCH IF INCORRECT.
110726  077203                    SOB     R2,1$

110730  020027  110766            CMP     R0,#FFBBF1+10     ;IS R0 CORRECT?
110734  001014                    BNE     FFB10             ;BRANCH IF INCORRECT.
110736  022705  000204            CMP     #204,R5           ;IS THE FPS CORRECT?
```

```
110742  001011                          BNE     FFB10           ;BRANCH IF INCORRECT.
110744  000411                          BR      FFBDONE

                                ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
110746  000177                  FFBTP1: 177
110750  167574                          167574
110752  137271                          137271
110754  107675                          107675
110756  177777                  FFBBF1: -1
110760  177777                          -1
110762  177777                          -1
110764  177777                          -1
110766                          FFB10:
110766  104000                          EMT                     ;
110770                          FFBDONE:
110770  004767  013576                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*************************************************************************
                                ;TEST 534       NEGF, ABSF AND TSTF SOURCE MODE 4 TEST
                                ;*************************************************************************
110774                          TS534:
```

```
110774  012700  111074                  MOV     #GGBTP1,R0              ;SET UP THE DATA BUFFER.
111000  012701  111104                  MOV     #GGBBF0,R1
111004  012702  000004                  MOV     #4,R2
111010  012021                  1$:     MOV     (R0)+,(R1)+
111012  077202                          SOB     R2,1$
111014  012700  000200                  MOV     #200,R0         ;SET FD.
111020  170100                          LDFPS   R0
111022  012700  111114                  MOV     #GGBBF1,R0      ;SET UP THE OPERAND ADDRESS.
111026  012737  111124  000004          MOV     #GGB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

111034  170640                  GGB2:   ABSD    -(R0)           ;TEST INSTRUCTION.

111036  170205                          STFPS   R5              ;GET FPS.
111040  012701  111104                  MOV     #GGBBF0,R1              ;CHECK RESULT.
111044  012702  000004                  MOV     #4,R2
111050  005721                  1$:     TST     (R1)+
111052  001024                          BNE     GGB10           ;BRANCH IF INCORRECT.
111054  077203                          SOB     R2,1$

111056  020027  111104                  CMP     R0,#GGBBF0      ;IS R0 CORRECT?
111062  001020                          BNE     GGB10           ;BRANCH IF INCORRECT.
111064  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
111070  001015                          BNE     GGB10           ;BRANCH IF INCORRECT.
111072  000415                          BR      GGBDONE

                                ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
111074  000177                  GGBTP1: 177
111076  117273                          117273
111100  147576                          147576
111102  177071                          177071
111104  177777                  GGBBF0: -1
111106  177777                          -1
111110  177777                          -1
111112  177777                          -1
111114  177777                  GGBBF1: -1
111116  177777                          -1
111120  177777                          -1
111122  177777                          -1
111124                          GGB10:
111124  104000                          EMT                     ;
111126                          GGBDONE:
111126  004767  013440                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 535       NEGF, ABSF AND TSTF SOURCE MODE 3 TEST
                                ;*******************************************************************************
111132                          TS535:

111132  012700  111232                  MOV     #HHBTP1,R0              ;SET UP THE DATA BUFFER.
111136  012701  111252                  MOV     #HHBBF0,R1
111142  012702  000010                  MOV     #10,R2
111146  012021                  1$:     MOV     (R0)+,(R1)+
```

```
111150  077202                                 SOB     R2,1$
111152  012700  000200                         MOV     #200,R0            ;SET FD.
111156  170100                                 LDFPS   R0
111160  012700  111262                         MOV     #HHBBF1,R0         ;SET UP THE OPERAND ADDRESS.
111164  012737  111272  000004                 MOV     #HHB10,@#ERRVECT   ;SET UP VECTOR 4 IN CASE OF AN ERROR.

111172  170630                         HHB2:   ABSD    @(R0)+             ;TEST INSTRUCTION.

111174  170205                                 STFPS   R5                 ;GET FPS.
111176  012701  111252                         MOV     #HHBBF0,R1                 ;CHECK RESULT.
111202  012702  000004                         MOV     #4,R2
111206  005721                         1$:     TST     (R1)+
111210  001030                                 BNE     HHB10              ;BRANCH IF INCORRECT.
111212  077203                                 SOB     R2,1$
111214  020027  111264                         CMP     R0,#HHBBF1+2       ;IS R0 CORRECT?
111220  001024                                 BNE     HHB10              ;BRANCH IF INCORRECT.
111222  022705  000204                         CMP     #204,R5            ;IS THE FPS CORRECT?
111226  001021                                 BNE     HHB10              ;BRANCH IF INCORRECT.
111230  000421                                 BR      HHBDONE

                                       ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
111232  000177                         HHBTP1: 177
111234  147576                                 147576
111236  177071                                 177071
111240  107576  111252  177777                 107576,HHBBF0,-1,-1,-1
111246  177777  177777
111252  177777                         HHBBF0: -1
111254  177777                                 -1
111256  177777                                 -1
111260  177777                                 -1
111262  177777                         HHBBF1: -1
111264  177777                                 -1
111266  177777                                 -1
111270  177777                                 -1
111272                                 HHB10:
111272  104000                                 EMT                        ;
111274                                 HHBDONE:
111274  004767  013272                         JSR     PC,.RSET           ;GO INITIALIZE THE FPS AND STACK; AND
                                                                          ;SEE IF THE USER HAS EXPRESSED
                                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                          ;THE USER TYPED CONTROL G?).
                                       ;*************************************************************************
                                       ;TEST 536        NEGF, ABSF AND TSTF SOURCE MODE 5 TEST
                                       ;*************************************************************************
111300                                 TS536:

111300  012700  111400                         MOV     #IIBTP1,R0                 ;SET UP THE DATA BUFFER.
111304  012701  111420                         MOV     #IIBBF0,R1
111310  012702  000010                         MOV     #10,R2
111314  012021                         1$:     MOV     (R0)+,(R1)+
111316  077202                                 SOB     R2,1$
111320  012700  000200                         MOV     #200,R0            ;SET FD.
111324  170100                                 LDFPS   R0
111326  012700  111432                         MOV     #IIBBF1+2,R0       ;SET UP THE OPERAND ADDRESS.
111332  012737  111440  000004                 MOV     #IIB10,@#ERRVECT   ;SET UP VECTOR 4 IN CASE OF AN ERROR.
```

```

111340  170750                    IIB2:   NEGD    @-(R0)          ;TEST INSTRUCTION.

111342  170205                            STFPS   R5              ;GET FPS.
111344  012701  111420                    MOV     #IIBBF0,R1                ;CHECK RESULT.
111350  012702  000004                    MOV     #4,R2
111354  005721                    1$:     TST     (R1)+
111356  001030                            BNE     IIB10           ;BRANCH IF INCORRECT.
111360  077203                            SOB     R2,1$
111362  020027  111430                    CMP     R0,#IIBBF1      ;IS R0 CORRECT?
111366  001024                            BNE     IIB10           ;BRANCH IF INCORRECT.
111370  022705  000204                    CMP     #204,R5         ;IS THE FPS CORRECT?
111374  001021                            BNE     IIB10           ;BRANCH IF INCORRECT.
111376  000421                            BR      IIBDONE


                                  ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
111400  000176                    IIBTP1: 176
111402  177074                            177074
111404  127374                            127374
111406  157677  111420  177777            157677,IIBBF0,-1,-1,-1
111414  177777  177777
111420  177777                    IIBBF0: -1
111422  177777                            -1
111424  177777                            -1
111426  177777                            -1
111430  177777                    IIBBF1: -1
111432  177777                            -1
111434  177777                            -1
111436  177777                            -1

111440                            IIB10:
111440  104000                            EMT                     ;
111442                            IIBDONE:
111442  004767  013124                    JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                  ;SEE IF THE USER HAS EXPRESSED
                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                  ;THE USER TYPED CONTROL G?).
                                  ;*************************************************************************
                                  ;TEST 537         NEGF, ABSF AND TSTF SOURCE MODE 6 TEST
                                  ;*************************************************************************
111446                            TS537:

111446  012700  111550                    MOV     #JJBTP1,R0                ;SET UP THE DATA BUFFER.
111452  012701  111562                    MOV     #JJBBF0,R1
111456  012702  000004                    MOV     #4,R2
111462  012021                    1$:     MOV     (R0)+,(R1)+
111464  077202                            SOB     R2,1$
111466  012700  000200                    MOV     #200,R0         ;SET FD.
111472  170100                            LDFPS   R0
111474  012700  111553                    MOV     #JJBBF0-7,R0    ;SET UP THE OPERAND ADDRESS.
111500  012737  111572  000004            MOV     #JJB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

111506  170660  000007            JJB2:   ABSD    7(R0)           ;TEST INSTRUCTION.

111512  170205                            STFPS   R5              ;GET FPS.
```

```
111514  012701  111562                  MOV     #JJBBF0,R1              ;CHECK RESULT.
111520  012702  000004                  MOV     #4,R2
111524  005721                  1$:     TST     (R1)+
111526  001021                          BNE     JJB10           ;BRANCH IF INCORRECT.
111530  077203                          SOB     R2,1$
111532  020027  111553                  CMP     R0,#JJBBF0-7    ;IS R0 CORRECT?
111536  001015                          BNE     JJB10           ;BRANCH IF INCORRECT.
111540  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
111544  001012                          BNE     JJB10           ;BRANCH IF INCORRECT.
111546  000412                          BR      JJBDONE

                                ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
111550  000177                  JJBTP1: 177
111552  161524                          161524
111554  131273                          131273
111556  107174  000000                  107174,
111562  177777                  JJBBF0: -1
111564  177777                          -1
111566  177777                          -1
111570  177777                          -1
111572                          JJB10:
111572  104000                          EMT                     ;
111574                          JJBDONE:
111574  004767  012772                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 540               NEGF, ABSF AND TSTF SOURCE MODE 7 TEST
                                ;*******************************************************************************
111600                          TS540:

111600  012700  111702                  MOV     #KKBTP1,R0              ;SET UP THE DATA BUFFER.
111604  012701  111722                  MOV     #KKBBF0,R1
111610  012702  000010                  MOV     #10,R2
111614  012021                  1$:     MOV     (R0)+,(R1)+
111616  077202                          SOB     R2,1$
111620  012700  000200                  MOV     #200,R0         ;SET FD.
111624  170100                          LDFPS   R0
111626  012700  111723                  MOV     #KKBBF1-7,R0    ;SET UP THE OPERAND ADDRESS.
111632  012737  111742  000004          MOV     #KKB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

111640  170770  000007          KKB2:   NEGD    @7(R0)          ;TEST INSTRUCTION.

111644  170205                          STFPS   R5              ;GET FPS.
111646  012701  111722                  MOV     #KKBBF0,R1              ;CHECK RESULT.
111652  012702  000004                  MOV     #4,R2
111656  005721                  1$:     TST     (R1)+
111660  001030                          BNE     KKB10           ;BRANCH IF INCORRECT.
111662  077203                          SOB     R2,1$
111664  020027  111723                  CMP     R0,#KKBBF1-7    ;IS R0 CORRECT?
111670  001024                          BNE     KKB10           ;BRANCH IF INCORRECT.
111672  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
111676  001021                          BNE     KKB10           ;BRANCH IF INCORRECT.
111700  000421                          BR      KKBDONE
```

```

                                        ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
111702  000177                          KKBTP1: 177
111704  167574                                  167574
111706  137271                                  137271
111710  107675  111722  177777                  107675,KKBBF0,-1,-1,-1
111716  177777  177777
111722  177777                          KKBBF0: -1
111724  177777                                  -1
111726  177777                                  -1
111730  177777                                  -1
111732  177777                          KKBBF1: -1
111734  177777                                  -1
111736  177777                                  -1
111740  177777                                  -1
111742                                  KKB10:
111742  104000                                  EMT                     ;
111744                                  KKBDONE:
111744  004767  012622                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).
                                        ;*****************************************************************************
                                        ;TEST 541       NEGF, ABSF AND TSTF SOURCE MODE 6, GR7, TEST
                                        ;*****************************************************************************
111750                                  TS541:
111750  012700  112040                          MOV     #LLBTP1,R0              ;SET UP THE DATA BUFFER.
111754  012701  112050                          MOV     #LLBBF0,R1
111760  012702  000004                          MOV     #4,R2
111764  012021                          1$:     MOV     (R0)+,(R1)+
111766  077202                                  SOB     R2,1$
111770  012700  000200                          MOV     #200,R0         ;SET FD.
111774  170100                                  LDFPS   R0
111776  012737  112060  000004                  MOV     #LLB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

112004  170767  000040                  LLB2:   NEGD    LLBBF0          ;TEST INSTRUCTION.

112010  170205                                  STFPS   R5              ;GET FPS.
112012  012701  112050                          MOV     #LLBBF0,R1              ;CHECK RESULT.
112016  012702  000004                          MOV     #4,R2
112022  005721                          1$:     TST     (R1)+
112024  001015                                  BNE     LLB10           ;BRANCH IF INCORRECT.
112026  077203                                  SOB     R2,1$
112030  022705  000204                          CMP     #204,R5         ;IS THE FPS CORRECT?
112034  001011                                  BNE     LLB10           ;BRANCH IF INCORRECT.
112036  000411                                  BR      LLBDONE

                                        ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
112040  000127                          LLBTP1: 127
112042  137475                                  137475
112044  147372                                  147372
112046  117057                                  117057
112050  177777                          LLBBF0: -1
112052  177777                                  -1
112054  177777                                  -1
```

```
112056  177777                                  -1

112060                          LLB10:
112060  104000                                  EMT                                     ;
112062                          LLBDONE:
112062  004767  012504                          JSR     PC,.RSET                        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                        ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 542               NEGF, ABSF AND TSTF SOURCE MODE 7, GR7, TEST
                                ;*******************************************************************************
112066                          TS542:

112066  012700  112156                          MOV     #MMBTP1,R0                              ;SET UP THE DATA BUFFER.
112072  012701  112176                          MOV     #MMBBF0,R1
112076  012702  000010                          MOV     #10,R2
112102  012021                          1$:     MOV     (R0)+,(R1)+
112104  077202                                  SOB     R2,1$
112106  012700  000200                          MOV     #200,R0                 ;SET FD.
112112  170100                                  LDFPS   R0
112114  012737  112216  000004                  MOV     #MMB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

112122  170677  000060          MMB2:   ABSD    @MMBBF1                 ;TEST INSTRUCTION.

112126  170205                                  STFPS   R5                      ;GET FPS.
112130  012701  112176                          MOV     #MMBBF0,R1                      ;CHECK RESULT.
112134  012702  000004                          MOV     #4,R2
112140  005721                          1$:     TST     (R1)+
112142  001025                                  BNE     MMB10                   ;BRANCH IF INCORRECT.
112144  077203                                  SOB     R2,1$
112146  022705  000204                          CMP     #204,R5                 ;IS THE FPS CORRECT?
112152  001021                                  BNE     MMB10                   ;BRANCH IF INCORRECT.
112154  000421                                  BR      MMBDONE

                                ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
112156  000137                  MMBTP1: 137
112160  045607                          045607
112162  101230                          101230
112164  045607  112176  177777          45607,MMBBF0,-1,-1,-1
112172  177777  177777
112176  177777                  MMBBF0: -1
112200  177777                          -1
112202  177777                          -1
112204  177777                          -1
112206  177777                  MMBBF1: -1
112210  177777                          -1
112212  177777                          -1
112214  177777                          -1

112216                          MMB10:
112216  104000                                  EMT                                     ;
112220                          MMBDONE:
112220  004767  012346                          JSR     PC,.RSET                        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                        ;SEE IF THE USER HAS EXPRESSED
```

```
                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                              ;THE USER TYPED CONTROL G?).
                            ;*****************************************************************
                            ;TEST 543       SPECIAL DEST, MODE 0, TEST
                            ;*****************************************************************
112224                      TS543:

112224  012700  000200              MOV     #200,R0           ;SET FD.
112230  170100                      LDFPS   R0
112232  012700  112312              MOV     #NNBTP1,R0                ;SET UP AC0.
112236  172410                      LDD     (R0),AC0
112240  170700              NNB2:   NEGD    AC0               ;TEST INSTRUCTION.

112242  170205                      STFPS   R5                ;GET FPS.
112244  012700  000200              MOV     #200,R0           ;SET FD.
112250  170100                      LDFPS   R0
112252  012700  112332              MOV     #NNBBF0,R0                ;GET THE RESULT.
112256  174010                      STD     AC0,(R0)
112260  012700  112332              MOV     #NNBBF0,R0                ;IS THE RESULT CORRECT?
112264  012701  112322              MOV     #NNBTP2,R1
112270  012702  000004              MOV     #4,R2
112274  022021              1$:     CMP     (R0)+,(R1)+
112276  001021                      BNE     NNB10             ;BRANCH IF INCORRECT.
112300  077203                      SOB     R2,1$
112302  022705  000210              CMP     #210,R5           ;IS THE FPS CORRECT?
112306  001015                      BNE     NNB10             ;BRANCH IF INCORRECT.
112310  000415                      BR      NNBDONE

                            ;THESE ARE DATA TABLES AND A DATA BUFFER.
112312  013572              NNBTP1: 013572
112314  046013                      46013
112316  057246                      57246
112320  013570                      013570
112322  113572              NNBTP2: 113572
112324  046013                      46013
112326  057246                      57246
112330  013570                      013570
112332  000000              NNBBF0: 0
112334  000000                      0
112336  000000                      0
112340  000000                      0

112342                      NNB10:
112342  104000                      EMT                       ;
112344                      NNBDONE:
112344  004767  012222              JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                              ;SEE IF THE USER HAS EXPRESSED
                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                              ;THE USER TYPED CONTROL G?).
                            ;*****************************************************************
                            ;TEST 544       SPECIAL DEST, MODE 1, TEST
                            ;*****************************************************************
112350                      TS544:
```

```
112350 012701 112452          MOV     #OOBTP1,R1        ;SET UP THE DATA BUFFER.
112354 012700 112462          MOV     #OOBTP2,R0
112360 012702 000004          MOV     #4,R2
112364 012021           1$:   MOV     (R0)+,(R1)+
112366 077202                 SOB     R2,1$
112370 012700 112452          MOV     #OOBTP1,R0
112374 042710 100000          BIC     #100000,(R0)      ;MAKE OPERAND POSITIVE.
112400 012701 000200          MOV     #200,R1           ;SET FD.
112404 170101                 LDFPS   R1

112406 170710           OOB2: NEGD    (R0)              ;TEST INSTRUCTION.
112410 170205                 STFPS   R5                ;GET FPS.
112412 012701 112452          MOV     #OOBTP1,R1                ;IS THE RESULT CORRECT.
112416 012702 112462          MOV     #OOBTP2,R2
112422 012703 000004          MOV     #4,R3
112426 022122           1$:   CMP     (R1)+,(R2)+
112430 001020                 BNE     OOB10             ;BRANCH IF INCORRECT.
112432 077303                 SOB     R3,1$
112434 022700 112452          CMP     #OOBTP1,R0        ;IS R0 CORRECT.
112440 001014                 BNE     OOB10             ;BRANCH IF INCORRECT.
112442 022705 000210          CMP     #210,R5           ;IS THE FPS CORRECT?
112446 001011                 BNE     OOB10             ;BRANCH IF INCORRECT.
112450 000411                 BR      OOBDONE

                        ;THESE ARE DATA TABLES AND A DATA BUFFER.
112452 023245           OOBTP1: 023245
112454 026720                 26720
112456 122324                 122324
112460 052672                 52672
112462 123245           OOBTP2: 123245
112464 026720                 26720
112466 122324                 122324
112470 052672                 52672

112472                  OOB10:
112472 104000                 EMT                       ;
112474                  OOBDONE:
112474 004767 012072          JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                        ;THE USER TYPED CONTROL G?).
                        ;*****************************************************************************
                        ;TEST 545       SPECIAL DEST, MODE 2, TEST
                        ;*****************************************************************************
112500                  TS545:

112500 012701 112602          MOV     #PPBTP1,R1        ;SET UP THE DATA BUFFER.
112504 012700 112612          MOV     #PPBTP2,R0
112510 012702 000004          MOV     #4,R2
112514 012021           1$:   MOV     (R0)+,(R1)+
112516 077202                 SOB     R2,1$
112520 012700 112602          MOV     #PPBTP1,R0
112524 042710 100000          BIC     #100000,(R0)      ;MAKE OPERAND POSITIVE.
112530 012701 000200          MOV     #200,R1           ;SET FD.
112534 170101                 LDFPS   R1
```

```
112536  170720                          PPB2:   NEGD    (R0)+               ;TEST INSTRUCTION.

112540  170205                                  STFPS   R5                  ;GET FPS.
112542  012701  112602                          MOV     #PPBTP1,R1                  ;IS THE RESULT CORRECT.
112546  012702  112612                          MOV     #PPBTP2,R2
112552  012703  000004                          MOV     #4,R3
112556  022122                          1$:     CMP     (R1)+,(R2)+
112560  001020                                  BNE     PPB10               ;BRANCH IF INCORRECT.
112562  077303                                  SOB     R3,1$
112564  022700  112612                          CMP     #PPBTP1+10,R0       ;IS R0 CORRECT.
112570  001014                                  BNE     PPB10               ;BRANCH IF INCORRECT.
112572  022705  000210                          CMP     #210,R5             ;IS THE FPS CORRECT?
112576  001011                                  BNE     PPB10               ;BRANCH IF INCORRECT.
112600  000411                                  BR      PPBDONE

                                        ;THESE ARE DATA TABLES AND A DATA BUFFER.
112602  023245                          PPBTP1: 023245
112604  026720                                  26720
112606  122324                                  122324
112610  052672                                  52672
112612  123245                          PPBTP2: 123245
112614  026720                                  26720
112616  122324                                  122324
112620  052672                                  52672

112622                                  PPB10:
112622  104000                                  EMT                         ;
112624                                  PPBDONE:
112624  004767  011742                          JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                            ;SEE IF THE USER HAS EXPRESSED
                                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                            ;THE USER TYPED CONTROL G?).
                                        ;*******************************************************************************
                                        ;TEST 546       SPECIAL DEST, MODE 4, TEST
                                        ;*******************************************************************************
112630                                  TS546:
112630  012701  112734                          MOV     #QQBTP1,R1          ;SET UP THE DATA BUFFER.
112634  012700  112754                          MOV     #QQBTP2,R0
112640  012702  000004                          MOV     #4,R2
112644  012021                          1$:     MOV     (R0)+,(R1)+
112646  077202                                  SOB     R2,1$
112650  012700  112744                          MOV     #QQBTP1+10,R0
112654  042760  100000  177770                  BIC     #100000,-10(R0) ;MAKE OPERAND POSITIVE.
112662  012701  000200                          MOV     #200,R1             ;SET FD.
112666  170101                                  LDFPS   R1

112670  170740                          QQB2:   NEGD    -(R0)               ;TEST INSTRUCTION.

112672  170205                                  STFPS   R5                  ;GET FPS.
112674  012701  112734                          MOV     #QQBTP1,R1                  ;IS THE RESULT CORRECT.
112700  012702  112754                          MOV     #QQBTP2,R2
112704  012703  000004                          MOV     #4,R3
112710  022122                          1$:     CMP     (R1)+,(R2)+
112712  001024                                  BNE     QQB10               ;BRANCH IF INCORRECT.
```

```
112714  077303                          SOB     R3,1$
112716  022700  112734                  CMP     #QQBTP1,R0          ;IS R0 CORRECT.
112722  001020                          BNE     QQB10               ;BRANCH IF INCORRECT.
112724  022705  000210                  CMP     #210,R5             ;IS THE FPS CORRECT?
112730  001015                          BNE     QQB10               ;BRANCH IF INCORRECT.
112732  000415                          BR      QQBDONE

                                ;THESE ARE DATA TABLES AND A DATA BUFFER.
112734  023245                  QQBTP1: 023245
112736  026720                          26720
112740  122324                          122324
112742  052672                          52672
112744  177777  177777  177777          .WORD   -1,-1,-1,-1
112752  177777
112754  123245                  QQBTP2: 123245
112756  026720                          26720
112760  122324                          122324
112762  052672                          52672

112764                          QQB10:
112764  104000                          EMT                         ;
112766                          QQBDONE:
112766  004767  011600                  JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                    ;SEE IF THE USER HAS EXPRESSED
                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                    ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 547       SPECIAL DEST, MODE 3, TEST
                                ;*******************************************************************************
112772                          TS547:

112772  012701  113102                  MOV     #RRBTP1,R1          ;SET UP THE DATA BUFFER.
112776  012700  113112                  MOV     #RRBTP2,R0
113002  012702  000004                  MOV     #4,R2
113006  012021                  1$:     MOV     (R0)+,(R1)+
113010  077202                          SOB     R2,1$
113012  012700  113122                  MOV     #RRBTP3,R0
113016  012710  113102                  MOV     #RRBTP1,(R0)
113022  042737  100000  113102          BIC     #100000,@#RRBTP1        ;MAKE THE OPERAND POSITIVE.
113030  012701  000200                  MOV     #200,R1             ;SET FD.
113034  170101                          LDFPS   R1

113036  170730                  RRB2:   NEGD    @(R0)+              ;TEST INSTRUCTION.

113040  170205                          STFPS   R5                  ;GET FPS.
113042  012701  113102                  MOV     #RRBTP1,R1                  ;IS THE RESULT CORRECT.
113046  012702  113112                  MOV     #RRBTP2,R2
113052  012703  000004                  MOV     #4,R3
113056  022122                  1$:     CMP     (R1)+,(R2)+
113060  001021                          BNE     RRB10               ;BRANCH IF INCORRECT.
113062  077303                          SOB     R3,1$
113064  022700  113124                  CMP     #RRBTP3+2,R0        ;IS R0 CORRECT.
113070  001015                          BNE     RRB10               ;BRANCH IF INCORRECT.
113072  022705  000210                  CMP     #210,R5             ;IS THE FPS CORRECT?
```

```
113076  001012                          BNE     RRB10           ;BRANCH IF INCORRECT.
113100  000412                          BR      RRBDONE

                                ;THESE ARE DATA TABLES AND A DATA BUFFER.
113102  023245                  RRBTP1: 023245
113104  026720                          26720
113106  122324                          122324
113110  052672                          52672
113112  123245                  RRBTP2: 123245
113114  026720                          26720
113116  123324                          123324
113120  052672                          52672
113122  113102                  RRBTP3: RRBTP1

113124                          RRB10:
113124  104000                          EMT                     ;
113126                          RRBDONE:
113126  004767  011440                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*******************************************************************************
                                ;TEST 550       SPECIAL DEST, MODE 5, TEST
                                ;*******************************************************************************
113132                          TS550:
113132  012701  113244                  MOV     #SSBTP1,R1      ;SET UP THE DATA BUFFER.
113136  012700  113254                  MOV     #SSBTP2,R0
113142  012702  000004                  MOV     #4,R2
113146  012021                  1$:     MOV     (R0)+,(R1)+
113150  077202                          SOB     R2,1$
113152  012700  113266                  MOV     #SSBTP3+2,R0
113156  012760  113244  177776          MOV     #SSBTP1,-2(R0)
113164  042737  100000  113244          BIC     #100000,@#SSBTP1        ;MAKE THE OPERAND POSITIVE.
113172  012701  000200                  MOV     #200,R1         ;SET FD.
113176  170101                          LDFPS   R1

113200  170750                  SSB2:   NEGD    @-(R0)          ;TEST INSTRUCTION.

113202  170205                          STFPS   R5              ;GET FPS.
113204  012701  113244                  MOV     #SSBTP1,R1              ;IS THE RESULT CORRECT.
113210  012702  113254                  MOV     #SSBTP2,R2
113214  012703  000004                  MOV     #4,R3
113220  022122                  1$:     CMP     (R1)+,(R2)+
113222  001021                          BNE     SSB10           ;BRANCH IF INCORRECT.
113224  077303                          SOB     R3,1$
113226  022700  113264                  CMP     #SSBTP3,R0      ;IS R0 CORRECT.
113232  001015                          BNE     SSB10           ;BRANCH IF INCORRECT.
113234  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
113240  001012                          BNE     SSB10           ;BRANCH IF INCORRECT.
113242  000412                          BR      SSBDONE

                                ;THESE ARE DATA TABLES AND A DATA BUFFER.
113244  023245                  SSBTP1: 023245
113246  026720                          26720
```

```
113250  122324                  122324
113252  052672                  52672
113254  123245          SSBTP2: 123245
113256  026270                  26270
113260  122324                  122324
113262  052672                  52672
113264  113244          SSBTP3: SSBTP1

113266                  SSB10:
113266  104000                  EMT                             ;
113270                  SSBDONE:
113270  004767  011276          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                        ;*******************************************************************************
                        ;TEST 551       SPECIAL DEST, FLOATING MODE 2, TEST
                        ;*******************************************************************************
113274                  TS551:
113274  012701  113376          MOV     #TTBTP1,R1              ;SET UP THE DATA BUFFER.
113300  012700  113406          MOV     #TTBTP2,R0
113304  012702  000004          MOV     #4,R2
113310  012021          1$:     MOV     (R0)+,(R1)+
113312  077202                  SOB     R2,1$
113314  012700  113376          MOV     #TTBTP1,R0
113320  042710  100000          BIC     #100000,(R0)            ;MAKE OPERAND POSITIVE.
113324  012701  000000          MOV     #000,R1                 ;SET FD.
113330  170101                  LDFPS   R1

113332  170720          TTB2:   NEGF    (R0)+                   ;TEST INSTRUCTION.

113334  170205                  STFPS   R5                      ;GET FPS.
113336  012701  113376          MOV     #TTBTP1,R1                      ;IS THE RESULT CORRECT.
113342  012702  113406          MOV     #TTBTP2,R2
113346  012703  000004          MOV     #4,R3
113352  022122          1$:     CMP     (R1)+,(R2)+
113354  001020                  BNE     TTB10                   ;BRANCH IF INCORRECT.
113356  077303                  SOB     R3,1$
113360  022700  113402          CMP     #TTBTP1+4,R0            ;IS R0 CORRECT.
113364  001014                  BNE     TTB10                   ;BRANCH IF INCORRECT.
113366  022705  000010          CMP     #010,R5                 ;IS THE FPS CORRECT?
113372  001011                  BNE     TTB10                   ;BRANCH IF INCORRECT.
113374  000411                  BR      TTBDONE

                        ;THESE ARE DATA TABLES AND A DATA BUFFER.
113376  023245          TTBTP1: 023245
113400  026720                  26720
113402  122324                  122324
113404  052672                  52672
113406  123245          TTBTP2: 123245
113410  026720                  26720
113412  122324                  122324
113414  052672                  52672

113416                  TTB10:
```

```
113416  104000                                  EMT                             ;
113420                          TTBDONE:
113420  004767  011146                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 552       SPECIAL DEST, MODE2, GR7 (IMMEDIATE), TEST
                                ;*******************************************************************************
113424                          TS552:
113424  012700  113542                          MOV     #UUBTP2,R0
113430  012701  113470                          MOV     #UUBTP1,R1              ;SET UP THE DATA BUFFER.
113434  012702  000004                          MOV     #4,R2
113440  012021                  1$:             MOV     (R0)+,(R1)+
113442  077202                                  SOB     R2,1$
113444  012700  113470                          MOV     #UUBTP1,R0
113450  042737  100000  113470                  BIC     #100000,@#UUBTP1                ;MAKE THE OPERAND POSITIVE.
113456  012701  000200                          MOV     #200,R1                 ;SET FD.
113462  170101                                  LDFPS   R1
113464  005001                                  CLR     R1

113466  170727                  UUB2:           NEGD    (R7)+                   ;TEST INSTRUCTION.
113470  005201  005201  005201  UUBTP1: 5201,5201,5201,5201
113476  005201

                                ;NOTE THAT AFTER EXECUTING THIS INSTRUCTION R1 SHOULD CONTAIN 3.
113500  170205                                  STFPS   R5                      ;GET FPS.
113502  012703  113470                          MOV     #UUBTP1,R3              ;IS THE RESULT CORRECT.
113506  012702  113542                          MOV     #UUBTP2,R2
113512  012704  000004                          MOV     #4,R4
113516  022322                  1$:             CMP     (R3)+,(R2)+
113520  001014                                  BNE     UUB10                   ;BRANCH IF INCORRECT.
113522  077403                                  SOB     R4,1$
113524  022701  000003                          CMP     #3,R1                   ;WAS R1 INCREMENTED CORRECTLY.
113530  001010                                  BNE     UUB10                   ;BRANCH IF INCORRECT.
113532  022705  000210                          CMP     #210,R5                 ;IS THE FPS CORRECT?
113536  001005                                  BNE     UUB10                   ;BRANCH IF INCORRECT.
113540  000405                                  BR      UUBDONE

                                ;THESE ARE DATA TABLE.
113542  105201                  UUBTP2: 105201
113544  005201                          5201
113546  005201                          5201
113550  005201                          5201

113552                          UUB10:
113552  104000                                  EMT                             ;
113554                          UUBDONE:
113554  004767  011012                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 553       SPECIAL DEST, MODE 6, TEST
                                ;*******************************************************************************
```

```
113560                                  TS553:
113560  012701  113674                          MOV     #XXBTP1,R1          ;SET UP THE DATA BUFFER.
113564  012700  113704                          MOV     #XXBTP2,R0
113570  012702  000004                          MOV     #4,R2
113574  012021                          1$:     MOV     (R0)+,(R1)+
113576  077202                                  SOB     R2,1$
113600  012700  106473                          MOV     #XXBTP1-5201,R0
113604  042737  100000  113674                  BIC     #100000,@#XXBTP1;MAKE OPERAND POSITIVE.
113612  012701  000200                          MOV     #200,R1             ;SET FD.
113616  170101                                  LDFPS   R1

113620  005001                                  CLR     R1
113622  170760  005201                  XXB2:   NEGD    5201(R0)                    ;TEST INSTRUCTION.

113626  170205                                  STFPS   R5                  ;GET FPS.
113630  005701                                  TST     R1
113632  001030                                  BNE     XXB10               ;WAS THE PC CORRECT AFTER EXECUTION?
113634  012701  113674                          MOV     #XXBTP1,R1                  ;IS THE RESULT CORRECT.
113640  012702  113704                          MOV     #XXBTP2,R2
113644  012703  000004                          MOV     #4,R3
113650  022122                          1$:     CMP     (R1)+,(R2)+
113652  001020                                  BNE     XXB10               ;BRANCH IF INCORRECT.
113654  077303                                  SOB     R3,1$
113656  022700  106473                          CMP     #XXBTP1-5201,R0     ;IS R0 CORRECT.
113662  001014                                  BNE     XXB10               ;BRANCH IF INCORRECT.
113664  022705  000210                          CMP     #210,R5             ;IS THE FPS CORRECT?
113670  001011                                  BNE     XXB10               ;BRANCH IF INCORRECT.
113672  000411                                  BR      XXBDONE


                                        ;THESE ARE DATA TABLES AND A DATA BUFFER.
113674  023245                          XXBTP1: 023245
113676  026720                                  26720
113700  122324                                  122324
113702  052672                                  52672
113704  123245                          XXBTP2: 123245
113706  026720                                  26720
113710  122324                                  122324
113712  052672                                  52672


113714                                  XXB10:
113714  104000                                  EMT                         ;
113716                                  XXBDONE:
113716  004767  010650                          JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                            ;SEE IF THE USER HAS EXPRESSED
                                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                            ;THE USER TYPED CONTROL G?).

                                        ;*************************************************************************
                                        ;TEST 554        SPECIAL DEST, MODE 7, TEST
                                        ;*************************************************************************
113722                                  TS554:

113722  012701  114044                          MOV     #YYBTP1,R1          ;SET UP THE DATA BUFFER.
113726  012700  114054                          MOV     #YYBTP2,R0
```

```
113732  012702  000004                  MOV     #4,R2
113736  012021                  1$:     MOV     (R0)+,(R1)+
113740  077202                          SOB     R2,1$
113742  012700  106663                  MOV     #YYBTP3-5201,R0
113746  012760  114044  005201          MOV     #YYBTP1,5201(R0)
113754  042737  100000  114044          BIC     #100000,@#YYBTP1        ;MAKE THE OPERAND POSITIVE.
113762  012701  000200                  MOV     #200,R1         ;SET FD.
113766  170101                          LDFPS   R1

113770  005001                          CLR     R1
113772  170770  005201          YYB2:   NEGD    @5201(R0)               ;TEST INSTRUCTION.

113776  170205                          STFPS   R5              ;GET FPS.
114000  005701                          TST     R1              ;WAS THE PC CORRECT AFTER EXECUTION?
114002  001031                          BNE     YYB10
114004  012701  114044                  MOV     #YYBTP1,R1              ;IS THE RESULT CORRECT.
114010  012702  114054                  MOV     #YYBTP2,R2
114014  012703  000004                  MOV     #4,R3
114020  022122                  1$:     CMP     (R1)+,(R2)+
114022  001021                          BNE     YYB10           ;BRANCH IF INCORRECT.
114024  077303                          SOB     R3,1$
114026  022700  106663                  CMP     #YYBTP3-5201,R0 ;IS R0 CORRECT.
114032  001015                          BNE     YYB10           ;BRANCH IF INCORRECT.
114034  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
114040  001012                          BNE     YYB10           ;BRANCH IF INCORRECT.
114042  000412                          BR      YYBDONE

                                ;THESE ARE DATA TABLES AND A DATA BUFFER.
114044  023245                  YYBTP1: 023245
114046  026720                          26720
114050  122324                          122324
114052  052672                          52672
114054  123245                  YYBTP2: 123245
114056  026720                          26720
114060  123324                          123324
114062  052672                          52672
114064  114044                  YYBTP3: YYBTP1
114066                          YYB10:
114066  104000                          EMT                     ;

114070                          YYBDONE:
114070  004767  010476                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 555               NEGD, ABSD AND TSTD TEST
                                ;*******************************************************************************
114074                          TS555:
                                ;TEST NEGD WITH POS NONZERO OPERAND
114074  004767  000526          WWB1:   JSR     PC,NATSUB
114100  000000                  1$:     0                       ;FLAG=NEGD.
114102  016341                  2$:     16341                   ;OPERAND.
114104  055772                          55772
114106  021133                          21133
```

```
114110  055447                  55447
114112  116341          3$:     116341                  ;RESULT.
114114  055772                  55772
114116  021133                  21133
114120  055447                  55447
114122  016341          4$:     16341                   ;ERROR RES.
114124  055772                  55772
114126  021133                  21133
114130  055447                  55447
114132  000207          5$:     207                     ;FPS BEFORE EXECUTION.
114134  000210                  210                     ;FPS AFTER EXECUTION.
114136  000200                  200                     ;ERROR FPS.
114140  177777                  -1                      ;FEC
                        ;TEST NEGD WITH NEG OPERAND.
114142  004767  000460  WWB2:   JSR     PC,NATSUB
114146  000000          1$:     0                       ;FLAG=NEGD.
114150  152525          2$:     152525                  ;OPERAND.
114152  053545                  53545
114154  055565                  55565
114156  057505                  57505
114160  052525          3$:     52525                   ;RESULT.
114162  053545                  53545
114164  055565                  55565
114166  057505                  57505
114170  152525          4$:     152525                  ;ERROR RES.
114172  053545                  53545
114174  055565                  55565
114176  057505                  57505
114200  000217          5$:     217                     ;FPS BEFORE EXECUTION.
114202  000200                  200                     ;FPS AFTER EXECUTION.
114204  000210                  210                     ;ERROR FPS.
114206  177777                  -1                      ;FEC
                        ;TEST ABSD WITH POSITIVE OPERAND
114210  004767  000412  WWB3:   JSR     PC,NATSUB
114214  000001          1$:     1                       ;FLAG=ABSD.
114216  060705          2$:     60705                   ;OPERAND.
114220  124735                  124735
114222  060124                  60124
114224  073560                  73560
114226  060705          3$:     60705                   ;RESULT.
114230  124735                  124735
114232  060124                  60124
114234  073560                  73560
114236  160705          4$:     160705                  ;ERROR RES.
114240  124735                  124735
114242  060124                  60124
114244  073560                  73560
114246  000217          5$:     217                     ;FPS BEFORE EXECUTION.
114250  000200                  200                     ;FPS AFTER EXECUTION.
114252  000210                  210                     ;ERROR FPS.
114254  177777                  -1                      ;EITHER BUT OP1B
                        ;TEST ABSD WITH NEG. OPERAND
114256  004767  000344  WWB4:   JSR     PC,NATSUB
114262  000001          1$:     1                       ;FLAG=ABSD.
114264  154345          2$:     154345                  ;OPERAND.
114266  076567                  76567
```

```
114270  032123                 32123
114272  043234                 43234
114274  054345         3$:     54345                        ;RESULT.
114276  076567                 76567
114300  032123                 32123
114302  043234                 43234
114304  154345         4$:     154345                       ;ERROR RES.
114306  076567                 76567
114310  032123                 32123
114312  043234                 43234
114314  000217         5$:     217                          ;FPS BEFORE EXECUTION.
114316  000200                 200                          ;FPS AFTER EXECUTION.
114320  177777                 -1                           ;ERROR FPS.
114322  177777                 -1
                       ;TEST WITH POSITIVE OP
114324  004767  000276 WWB5:   JSR     PC,NATSUB
114330  000002         1$:     2                            ;FLAG=TSTD.
114332  012321         2$:     12321                        ;OPERAND.
114334  045654                 45654
114336  070107                 70107
114340  034543                 34543
114342  012321         3$:     12321                        ;RESULT.
114344  045654                 45654
114346  070107                 70107
114350  034543                 34543
114352  112321         4$:     112321                       ;ERROR RES.
114354  045654                 45654
114356  070107                 70107
114360  034543                 34543
114362  000217         5$:     217                          ;FPS BEFORE EXECUTION.
114364  000200                 200                          ;FPS AFTER EXECUTION.
114366  000210                 210                          ;ERROR FPS.
114370  177777                 -1
                       ;TEST TSTD WITH NEG OP
114372  004767  000230 WWB6:   JSR     PC,NATSUB
114376  000002         1$:     2                            ;FLAG=TSTD.
114400  123765         2$:     123765                       ;OPERAND.
114402  023407                 23407
114404  034510                 34510
114406  045621                 45621
114410  123765         3$:     123765                       ;RESULT.
114412  023407                 23407
114414  034510                 34510
114416  045621                 45621
114420  023765         4$:     23765                        ;ERROR RES.
114422  023407                 23407
114424  034510                 34510
114426  045621                 45621
114430  000207         5$:     207                          ;FPS BEFORE EXECUTION.
114432  000210                 210                          ;FPS AFTER EXECUTION.
114434  000200                 200                          ;ERROR FPS.
114436  177777                 -1
                       ;TEST TSTD 0 OP
114440  004767  000162 WWB7:   JSR     PC,NATSUB
114444  000002         1$:     2                            ;FLAG=TSTD.
114446  000175         2$:     175                          ;OPERAND.
```

```
114450  176737                  176737
114452  071727                  71727
114454  037574                  37574
114456  000175          3$:     175                     ;RESULT.
114460  176737                  176737
114462  071727                  71727
114464  037574                  37574
114466  000000          4$:     0                       ;ERROR RES.
114470  000000                  0
114472  000000                  0
114474  000000                  0
114476  000200          5$:     200                     ;FPS BEFORE EXECUTION.
114500  000204                  204                     ;FPS AFTER EXECUTION.
114502  000214                  214                     ;ERROR FPS.
114504  177777                  -1
                        ;TEST TSTD -0 OP FIUV=0
114506  004767  000114  WWB10:  JSR     PC,NATSUB
114512  000002          1$:     2                       ;FLAG=TSTD.
114514  100123          2$:     100123                  ;OPERAND.
114516  021012                  21012
114520  034565                  34565
114522  043210                  43210
114524  100123          3$:     100123                  ;RESULT.
114526  021012                  21012
114530  034565                  34565
114532  043210                  43210
114534  000000          4$:     0                       ;ERROR RES.
114536  000000                  0
114540  000000                  0
114542  000000                  0
114544  040203          5$:     40203                   ;FPS BEFORE EXECUTION.
114546  040214                  040214                  ;FPS AFTER EXECUTION.
114550  140214                  140214                  ;ERROR FPS.
114552  177777                  -1
                        ;TEST TSTD -0 OP FIUV=1
114554  004767  000046  WWB11:  JSR     PC,NATSUB
114560  000002          1$:     2                       ;FLAG=TSTD.
114562  100137          2$:     100137                  ;OPERAND.
114564  024613                  24613
114566  057024                  57024
114570  060137                  60137
114572  100137          3$:     100137                  ;RESULT.
114574  024613                  24613
114576  057024                  57024
114600  060137                  60137
114602  000000          4$:     0                       ;ERROR RES.
114604  000000                  0
114606  000000                  0
114610  000000                  0
114612  044200          5$:     44200                   ;FPS BEFORE EXECUTION.
114614  144214                  144214                  ;FPS AFTER EXECUTION.
114616  044214                  044214                  ;ERROR FPS.
114620  000014                  14
114622  000167  000162          JMP     WWBDONE

                        ;THIS SUBROUTINE, NATSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
```

```
                          ;THE EITHER A TSTD, AN ABSD OR A NEGD INSTRUCTION AND CHECK THE RESULTS. A CALL
                          ;TO IT IS MADE THUS:
                          ;
                          ;               JSR     PC,@#NATSUB
                          ;               FLAG:   .WORD   X               ;INSTRUCTION TYPE FLAG.
                          ;               ACARG:  .WORD   X,X,X,X         ;OPERAND
                          ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
                          ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
                          ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                          ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                          ;               FEC:    .WORD   X               ;EXPECTED FEC
                          ;               ERFPS:  .WORD   X               ;ERROR FPS.
                          ;               ERR1:   ERROR   X               ;DATA ERROR.
                          ;                       BR      CONT
                          ;               ERR2:   ERROR   X               ;FPS ERROR.
                          ;               CONT:                           ;RETURN ADDRESS

                          ;THE OPERAND IS SET UP IN NATBF1. THEN
                          ;THE EITHER THE TSTD, NEGD OR ABSD INSTRUCTION IS EXECUTED.
                          ;NATSUB USES THE FIRST OPERAND AS A FLAG TO DETERMINE WHICH INSTRUCTION
                          ;IS TO BE EXECUTED: 0 = NEGD, 1 = ABSD, 2 = TSTD.
                          ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                          ;COMPARED WITH FPSA. IF THIS TOO IS CORRECT NATSUB RETURNS CONTROL
                          ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD NATSUB
                          ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN NATSUB WILL RETURN
                          ;TO THE ERROR CALL AT ERR2, OTHERWISE NATSUB ITSELF
                          ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                          ;INSTRUCTION IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                          ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                          ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN NATSUB
                          ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                          ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND NATSUB WILL
                          ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.


114626 012601             NATSUB: MOV     (SP)+,R1                ;GET A POINTER TO THE ARGUMENTS.
114630 010102                     MOV     R1,R2           ;COPY THE OPERAND.
114632 062702 000002              ADD     #2,R2
114636 012703 114776              MOV     #NATBF1,R3
114642 012704 000004              MOV     #4,R4
114646 012223             1$:     MOV     (R2)+,(R3)+
114650 077402                     SOB     R4,1$
114652 016100 000032              MOV     32(R1),R0       ;LOAD THE FPS.
114656 170100                     LDFPS   R0
114660 012700 114776              MOV     #NATBF1,R0      ;SET UP THE OPERAND ADDRESS.
114664 011102                     MOV     (R1),R2         ;GET THE FLAG TO DETERMINE WHICH
114666 006302                     ASL     R2              ;INSTRUCTION TO EXECUTE.
114670 006302                     ASL     R2              ;0 = NEGD, 1 = ABSD, 2 = TSTD
114672 012703 114702              MOV     #NATINS,R3
114676 060203                     ADD     R2,R3
114700 000113                     JMP     (R3)            ;GO EXECUTE THE INSTRUCTION.
114702 170710             NATINS: NEGD    (R0)
114704 000403                     BR      2$
114706 170610                     ABSD    (R0)
114710 000401                     BR      2$
114712 170510                     TSTD    (R0)
```

```
114714  170204                  2$:     STFPS   R4                      ;GET THE FPS.
114716  170305                          STST    R5                      ;GET THE FEC.
114720  010100                          MOV     R1,R0                   ;WAS THE RESULT CORRECT?
114722  062700  000012                  ADD     #12,R0
114726  012702  114776                  MOV     #NATBF1,R2
114732  012703  000004                  MOV     #4,R3
114736  022022                  3$:     CMP     (R0)+,(R2)+
114740  001014                          BNE     10$                     ;BRANCH IF INCORRECT.
114742  077303                          SOB     R3,3$
114744  026104  000034                  CMP     34(R1),R4               ;WAS THE FPS CORRECT?
114750  001010                          BNE     10$                     ;BRANCH IF INCORRECT.
114752  005761  000034                  TST     34(R1)                  ;IF THE EXPECTED FPS WAS NEGATIVE CHECK THE FEC.
114756  100003                          BPL     4$
114760  026105  000040                  CMP     40(R1),R5               ;WAS THE FEC CORRECT.
114764  001002                          BNE     10$                     ;BRANCH IF INCORRECT.
114766  000161  000042          4$:     JMP     42(R1)                  ;RETURN.

114772                          10$:
114772  104000                          EMT                             ;

114774  177777                          .WORD   -1
114776  177777  177777  177777  NATBF1: .WORD   -1,-1,-1,-1,-1
115004  177777  177777

115010                          WWBDONE:
115010  004767  007556                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).



                                ;*************************************************************************
                                ;TEST 556        SOURCE MODES, MODE 1 (FL=0), TEST
                                ;*************************************************************************
115014                          TS556:



115014  012700  115064                  MOV     #AACTP1,R0              ;SET UP TEST DATA IN BUFFER.
115020  012710  147517                  MOV     #147517,(R0)
115024  012737  115040  037364          MOV     #AAC2,@#$TMP2
115032  012737  115070  000004          MOV     #AAC10,@#ERRVECT                ;SET UP FOR TRAPS TO 4.
115040  170110                  AAC2:   LDFPS   (R0)                    ;TEST INSTRUCTION.

115042  170205                          STFPS   R5                      ;GET FPS

115044  020027  115064                  CMP     R0,#AACTP1              ;IS R0 CORRECT?
115050  001007                          BNE     AAC10                   ;BR IF NOT.
115052  022705  147517                  CMP     #147517,R5              ;IS FPS CORRECT?
115056  001004                          BNE     AAC10                   ;BR IF NOT.
115060  000404                          BR      AACDONE

                                ;TEST BUFFER AND DATA:
```

```
115062  177777                          -1
115064  147517                  AACTP1: 147517
115066  177777                          -1
115070                          AAC10:
115070  104000                          EMT                             ;

115072                          AACDONE:
115072  004767  007474                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************************
                                ;TEST 557       SOURCE MODES, MODE 2 (FL=0), TEST
                                ;*******************************************************************************
115076                          TS557:


115076  012700  115140                  MOV     #BBCTP1,R0      ;SET UP TEST DATA IN BUFFER.
115102  012710  145212                  MOV     #145212,(R0)
115106  012737  115144  000004          MOV     #BBC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.

115114  170120                  BBC2:   LDFPS   (R0)+           ;TEST INSTRUCTION.

115116  170205                          STFPS   R5              ;GET FPS

115120  020027  115142                  CMP     R0,#BBCTP1+2    ;IS R0 CORRECT?
115124  001007                          BNE     BBC10           ;BR IF NOT.
115126  022705  145212                  CMP     #145212,R5      ;IS THE FPS CORRECT?
115132  001004                          BNE     BBC10           ;BR IF NOT.
115134  000404                          BR      BBCDONE


                                ;TEST BUFFER AND DATA:
115136  177777                          -1
115140  177777                  BBCTP1: .WORD   -1
115142  177777                          -1
```

```
115144                          BBC10:
115144  104000                          EMT                             ;
115146                          BBCDONE:
115146  004767  007420                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).


                                ;*******************************************************************************
                                ;TEST 560       SOURCE MODES, MODE 4 (FL=0), TEST
                                ;*******************************************************************************
115152                          TS560:


115152  012700  115234                  MOV     #DDCTP1+2,R0            ;SET UP THE TEST DATA BUFFER.
115156  012760  105252  177776          MOV     #105252,-2(R0)
115164  012737  115200  037364          MOV     #DDC2,@#$TMP2
115172  012737  115244  000004          MOV     #DDC10,@#ERRVEC
115200  170140                  DDC2:   LDFPS   -(R0)
115202  170205                          STFPS   R5
115204  020027  115232                  CMP     R0,#DDCTP1
115210  001015                          BNE     DDC10
115212  022705  105252                  CMP     #105252,R5
115216  001012                          BNE     DDC10
115220  000412                          BR      DDCDONE

115222  177777  177777  177777          -1,-1,-1,-1
115230  177777
115232  177777                  DDCTP1: -1
115234  177777  177777  177777          -1,-1,-1,-1
115242  177777
115244                          DDC10:
115244  104000                          EMT                             ;
115246                          DDCDONE:
115246  004767  007320                  JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).
                                ;*******************************************************************************
                                ;TEST 561       SOURCE MODES, MODE 3 (FL=0), TEST
                                ;*******************************************************************************
115252                          TS561:
115252  012700  115340                  MOV     #EECTP2,R0
115256  012710  115330                  MOV     #EECTP1,(R0)
115262  012767  103456  000040          MOV     #103456,EECTP1
115270  012737  115352  000004          MOV     #EEC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
115276  170130                  EEC2:   LDFPS   @(R0)+                  ;TEST INSTRUCTION.
115300  170205                          STFPS   R5                      ;GET THE FPS.
115302  020027  115342                  CMP     R0,#EECTP2+2            ;IS R0 CORRECT?
115306  001021                          BNE     EEC10                   ;BR IF NOT.
115310  022705  103456                  CMP     #103456,R5              ;IS THE FPS CORRECT?
115314  001016                          BNE     EEC10                   ;BR IF NOT.
```

```
115316  000416                                  BR      EECDONE


                                ;TEST BUFFER AND DATA:
115320  177777  177777  177777                  -1,-1,-1,-1
115326  177777
115330  177777                  EECTP1: -1
115332  177777  177777  177777                  -1,-1,-1
115340  115330  177777  177777  EECTP2: EECTP1,-1,-1,-1,
115346  177777  000000

115352                          EEC10:
115352  104000                                  EMT                             ;
115354                          EECDONE:
115354  004767  007212                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).
                                ;*****************************************************************************
                                ;TEST 562       SOURCE MODES, MODE 5 (FL=0), TEST
                                ;*****************************************************************************
115360                          TS562:
115360  012700  115444                          MOV     #FFCTP2+2,R0            ;SET UP THE TEST DATA BUFFER.
115364  012760  115432  177776                  MOV     #FFCTP1,-2(R0)
115372  012737  045412  115432                  MOV     #45412,@#FFCTP1
115400  012737  115452  000004                  MOV     #FFC10,@#ERRVECT        ;SET UP FOR TRAPS TO 4.
115406  170150                  FFC2:   LDFPS   @-(R0)                          ;TEST INSTRUCTION.
115410  170205                          STFPS   R5                              ;GET THE FPS.
115412  020027  115442                          CMP     R0,#FFCTP2              ;IS R0 CORRECT?
115416  001015                                  BNE     FFC10                   ;BR IF NOT.
115420  022705  045412                          CMP     #45412,R5               ;IS THE FPS CORRECT?
115424  001012                                  BNE     FFC10                   ;BR IF NOT.
115426  000412                                  BR      FFCDONE


                                ;TEST BUFFER AND DATA:
115430  177777                                  -1
115432  177777                  FFCTP1: -1
115434  177777  177777  177777                  -1,-1,-1
115442  115432  177777  177777  FFCTP2: FFCTP1,-1,-1,-1
115450  177777

115452                          FFC10:
115452  104000                                  EMT                             ;
115454                          FFCDONE:
115454  004767  007112                          JSR     PC,.RSET                ;GO INITIALIZE THE FPS AND STACK; AND
                                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                                ;THE USER TYPED CONTROL G?).
                                ;*****************************************************************************
                                ;TEST 563       SOURCE MODES, MODE 6 (FL=0), TEST
                                ;*****************************************************************************
115460                          TS563:
115460  012700  110333                          MOV     #GGCTP1-5201,R0 ;SET UP THE TEST DATA BUFFER.
```

```
115464  012737  046543  115534                  MOV     #46543,@#GGCTP1
115472  005001                                  CLR     R1
115474  012737  115546  000004                  MOV     #GGC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
115502  170160  005201          GGC2:           LDFPS   5201(R0)                 ;TEST INSTRUCTION.
115506  170204                                  STFPS   R4               ;GET THE FPS.
115510  005701                                  TST     R1               ;WAS PC CORRECT AFTER EXECUTION?
115512  001015                                  BNE     GGC10            ;BR IF NOT.
115514  020027  110333                          CMP     R0,#GGCTP1-5201  ;IS R0 CORRECT?
115520  001012                                  BNE     GGC10            ;BR IF NOT.
115522  022704  046543                          CMP     #46543,R4        ;IS THE FPS CORRECT?
115526  001007                                  BNE     GGC10            ;BR IF NOT.
115530  000407                                  BR      GGCDONE



                                ;TEST BUFFER AND DATA:
115532  177777                                  -1
115534  177777  177777  177777  GGCTP1: -1,-1,-1,-1
115542  177777
115544  177777                                  -1
115546                          GGC10:
115546  104000                                  EMT                      ;
115550                          GGCDONE:
115550  004767  007016                          JSR     PC,.RSET         ;GO INITIALIZE THE FPS AND STACK; AND
                                                                         ;SEE IF THE USER HAS EXPRESSED
                                                                         ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                         ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                         ;THE USER TYPED CONTROL G?).
                                ;*********************************************************************
                                ;TEST 564       SOURCE MODES, MODE 7 (FL=0), TEST
                                ;*********************************************************************
115554                          TS564:
115554  012700  110445                          MOV     #HHCTP2-5201,R0 ;SET UP THE TEST DATA BUFFER.
115560  012760  115636  005201                  MOV     #HHCTP1,5201(R0)
115566  012737  004547  115636                  MOV     #4547,@#HHCTP1
115574  005001                                  CLR     R1
115576  012737  115656  000004                  MOV     #HHC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
115604  170170  005201          HHC2:           LDFPS   @5201(R0)                ;TEST INSTRUCTION.
115610  170204                                  STFPS   R4               ;GET THE FPS.
115612  005701                                  TST     R1               ;WAS PC CORRECT AFTER EXECUTION?
115614  001020                                  BNE     HHC10            ;BR IF NOT.
115616  020027  110445                          CMP     R0,#HHCTP2-5201  ;IS R0 CORRECT?
115622  001015                                  BNE     HHC10            ;BR IF NOT.
115624  022704  004547                          CMP     #4547,R4         ;IS THE FPS CORRECT?
115630  001012                                  BNE     HHC10            ;BR IF NOT.
115632  000412                                  BR      HHCDONE



                                ;TEST BUFFER AND DATA:
115634  177777                                  -1
115636  177777  177777  177777  HHCTP1: .WORD -1,-1,-1,-1
115644  177777
115646  177777  177777  177777  HHCTP2: .WORD -1,-1,-1,-1
115654  177777
115656                          HHC10:
115656  104000                                  EMT                      ;
115660                          HHCDONE:
```

```
115660  004767  006706                 JSR     PC,.RSET              ;GO INITIALIZE THE FPS AND STACK; AND
                                                                     ;SEE IF THE USER HAS EXPRESSED
                                                                     ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                     ;THE USER TYPED CONTROL G?).



                               ;*******************************************************************************
                               ;TEST 565          SOURCE MODES, MODE 2 GR7 (FL=1), TEST
                               ;*******************************************************************************
115664                         TS565:

115664  012737  115722  000004         MOV     #IIC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
115672  012700  000300                 MOV     #300,R0
115676  170100                         LDFPS   R0
115700  005001                         CLR     R1

115702  177027                 IIC2:   LDCLD   (R7)+,AC0                     ;TEST INSTRUCTION.
115704  005201                         5201
115706  005201                         5201
115710  005201                         5201
115712  005201                         5201

115714  020127  000003                 CMP     R1,#3                 ;WAS PC CORRECT AFTER EXECUTION?
115720  001401                         BEQ     IICDONE
115722                         IIC20:
115722  104000                         EMT                           ;

115724                         IICDONE:
115724  004767  006642                 JSR     PC,.RSET              ;GO INITIALIZE THE FPS AND STACK; AND
                                                                     ;SEE IF THE USER HAS EXPRESSED
                                                                     ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                     ;THE USER TYPED CONTROL G?).



                               ;*******************************************************************************
                               ;TEST 566          SOURCE MODES, MODE 2 (FL=1), TEST
                               ;*******************************************************************************
115730                         TS566:

115730  012700  000300                 MOV     #300,R0
115734  170100                         LDFPS   R0
115736  012700  116002                 MOV     #TCCBF0,R0            ;SET UP THE TEST DATA BUFFER.
115742  177020                 TCC2:   LDCLD   (R0)+,AC0                     ;TEST INSTRUCTION.

115744  170204                         STFPS   R4                    ;GET THE FPS.
115746  012701  116012                 MOV     #TCCBF1,R1            ;GET THE RESULT.
115752  012702  000200                 MOV     #200,R2
115756  170102                         LDFPS   R2
115760  174011                         STD     AC0,(R1)
115762  020027  116006                 CMP     R0,#TCCBF0+4          ;IS R0 CORRECT?
115766  001401                         BEQ     TCC3
115770  104000                         EMT                           ;
```

```
115772  022704  000300          TCC3:   CMP     #300,R4 ;IS THE FPS CORRECT?
115776  001411                          BEQ     TCCDONE
116000  104000                          EMT                             ;


                                ;TEST BUFFER AND DATA:
116002  001234  067076  054321  TCCBF0: .WORD   01234,67076,54321,012345
116010  012345
116012  177777  177777  177777  TCCBF1: -1,-1,-1,-1
116020  177777

116022                          TCCDONE:
116022  004767  006544                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).



                                ;*******************************************************************************
                                ;TEST 567       LDCIF AND LDCLF TEST
                                ;*******************************************************************************
116026                          TS567:


                                ;ZERO   OPERAND FL=0

116026  004737  116720          KKC1:   JSR     PC,@#LDCFSUB    ;GO EXECUTE INSTRUCTION.

116032  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
116036  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
116042  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
116046  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
116050  000004                          4                       ;FPS AFTER EXECUTION.
116052  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;ZERO   OPERAND FL=0

116054  004737  116720          KKC2:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.

116060  000000  177777          1$:     .WORD   0,-1            ;FSRC OPERAND.
116064  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
116070  004177  177400          3$:     4177,177400             ;ANTICIPATED ERRONEOUS RESULT.
116074  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
116076  000004                          4                       ;FPS AFTER EXECUTION.
116100  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;ZERO   OPERAND FL=1

116102  004737  116720          KKC3:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.

116106  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
116112  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
116116  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
116122  000100                  4$:     100                     ;FPS BEFORE EXECUTION.
116124  000104                          104                     ;FPS AFTER EXECUTION.
116126  000004                          4                       ;ANTICIPATED ERRONEOUS FPS.
```

```
                                ;OPERAND        POSITIVE        FL=0
116130  004737  116720          KKC4:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116134  040000  000000          1$:     .WORD   40000,0         ;FSRC OPERAND.
116140  043600  000000          2$:     .WORD   43600,0         ;EXPECTED RESULT.
116144  047600  000000          3$:     .WORD   47600,0         ;ANTICIPATED ERRONEOUS RESULT.
116150  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
116152  000000                          0                       ;FPS AFTER EXECUTION.
116154  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=1,     FL=0
116156  004737  116720          KKC5:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116162  000001  000000          1$:     .WORD   1,0             ;FSRC OPERAND.
116166  040200  000000          2$:     .WORD   40200,0         ;EXPECTED RESULT.
116172  044200  000000          3$:     .WORD   44200,0         ;ANTICIPATED ERRONEOUS RESULT.
116176  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
116200  000000                          0                       ;FPS AFTER EXECUTION.
116202  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=       PATTERN FL=0
116204  004737  116720          KKC6:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116210  000252  000000          1$:     .WORD   252,0           ;FSRC OPERAND.
116214  042052  000000          2$:     .WORD   42052,0         ;EXPECTED RESULT.
116220  046052  000000          3$:     .WORD   46052,0         ;ANTICIPATED ERRONEOUS RESULT.
116224  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
116226  000000                          0                       ;FPS AFTER EXECUTION.
116230  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=-40000 FL=0
116232  004737  116720          KKC7:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116236  140000  000000          1$:     .WORD   -40000,0                ;FSRC OPERAND.
116242  143600  000000          2$:     .WORD   143600,0                ;EXPECTED RESULT.
116246  043600  000000          3$:     .WORD   43600,0         ;ANTICIPATED ERRONEOUS RESULT.
116252  000007                  4$:     7                       ;FPS BEFORE EXECUTION.
116254  000010                          10                      ;FPS AFTER EXECUTION.
116256  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=-1     FL-0
116260  004737  116720          KKC10:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116264  177777  000000          1$:     .WORD   -1,0            ;FSRC OPERAND.
116270  140200  000000          2$:     .WORD   140200,0                ;EXPECTED RESULT.
116274  144000  000400          3$:     .WORD   144000,400              ;ANTICIPATED ERRONEOUS RESULT.
116300  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
116302  000010                          10                      ;FPS AFTER EXECUTION.
116304  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=PATTERN        FL=0
116306  004737  116720          KKC11:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116312  125252  000000          1$:     .WORD   125252,0                ;FSRC OPERAND.
116316  143652  126000          2$:     .WORD   143652,126000           ;EXPECTED RESULT.
116322  043652  126000          3$:     .WORD   43652,126000            ;ANTICIPATED ERRONEOUS RESULT.
116326  000007                  4$:     7                       ;FPS BEFORE EXECUTION.
116330  000010                          10                      ;FPS AFTER EXECUTION.
116332  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND        POS     FL-1
116334  004737  116720          KKC12:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
116340  040000  000000          1$:     .WORD   40000,0         ;FSRC OPERAND.
116344  047600  000000          2$:     .WORD   47600,0         ;EXPECTED RESULT.
116350  043600  000000          3$:     .WORD   43600,0         ;ANTICIPATED ERRONEOUS RESULT.
116354  000117                  4$:     117                     ;FPS BEFORE EXECUTION.
116356  000100                          100                     ;FPS AFTER EXECUTION.
116360  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
```

```
                                    ;OPERAND=1           FL=1
116362  004737  116720              KKC13:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116366  000000  000001              1$:     .WORD   0,1               ;FSRC OPERAND.
116372  040200  000000              2$:     .WORD   40200,0           ;EXPECTED RESULT.
116376  034200  000000              3$:     .WORD   34200,0           ;ANTICIPATED ERRONEOUS RESULT.
116402  000100                      4$:     100                       ;FPS BEFORE EXECUTION.
116404  000100                              100                       ;FPS AFTER EXECUTION.
116406  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=            PATTERN FL=1
116410  004737  116720              KKC14:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116414  000000  000252              1$:     .WORD   0,252             ;FSRC OPERAND.
116420  042052  000000              2$:     .WORD   42052,0           ;EXPECTED RESULT.
116424  036052  000000              3$:     .WORD   36052,0           ;ANTICIPATED ERRONEOUS RESULT.
116430  000111                      4$:     111                       ;FPS BEFORE EXECUTION.
116432  000100                              100                       ;FPS AFTER EXECUTION.
116434  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=-40000,0          FL=1
116436  004737  116720              KKC15:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116442  140000  000000              1$:     .WORD   -40000,0                  ;FSRC OPERAND.
116446  147600  000000              2$:     .WORD   147600,0                  ;EXPECTED RESULT.
116452  047600  000000              3$:     .WORD   47600,0           ;ANTICIPATED ERRONEOUS RESULT.
116456  000107                      4$:     107                       ;FPS BEFORE EXECUTION.
116460  000110                              110                       ;FPS AFTER EXECUTION.
116462  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=-1,-1   FL=1
116464  004737  116720              KKC16:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116470  177777  177777              1$:     .WORD   -1,-1             ;FSRC OPERAND.
116474  140200  000000              2$:     .WORD   140200,0                  ;EXPECTED RESULT.
116500  150000  000000              3$:     .WORD   150000,0                  ;ANTICIPATED ERRONEOUS RESULT.
116504  000100                      4$:     100                       ;FPS BEFORE EXECUTION.
116506  000110                              110                       ;FPS AFTER EXECUTION.
116510  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=-PATTERN          FL=1,  ROUND MODE
116512  004737  116720              KKC17:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116516  125252  125252              1$:     .WORD   125252,125252             ;FSRC OPERAND.
116522  147652  125253              2$:     .WORD   147652,125253             ;EXPECTED RESULT.
116526  047652  125253              3$:     .WORD   47652,125253              ;ANTICIPATED ERRONEOUS RESULT.
116532  000105                      4$:     105                       ;FPS BEFORE EXECUTION.
116534  000110                              110                       ;FPS AFTER EXECUTION.
116536  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=77777,177500    FL=1,  ROUND MODE
116540  004737  116720              KKC20:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116544  077777  177500              1$:     .WORD   77777,177500              ;FSRC OPERAND.
116550  047777  177777              2$:     .WORD   47777,177777              ;EXPECTED RESULT.
116554  047777  177776              3$:     .WORD   47777,177776              ;ANTICIPATED ERRONEOUS RESULT.
116560  000117                      4$:     117                       ;FPS BEFORE EXECUTION.
116562  000100                              100                       ;FPS AFTER EXECUTION.
116564  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
                                    ;OPERAND=40000,000100    FL=1,  ROUND MODE
116566  004737  116720              KKC21:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116572  040000  000100              1$:     .WORD   40000,100                 ;FSRC OPERAND.
116576  047600  000001              2$:     .WORD   47600,1           ;EXPECTED RESULT.
116602  047600  000000              3$:     .WORD   47600,0           ;ANTICIPATED ERRONEOUS RESULT.
116606  000102                      4$:     102                       ;FPS BEFORE EXECUTION.
116610  000100                              100                       ;FPS AFTER EXECUTION.
116612  177777                              -1                        ;ANTICIPATED ERRONEOUS FPS.
```

```
                                ;OPERAND=40000,000100    FL=1,    TRUNC MODE
116614  004737  116720          KKC22:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116620  040000  000100          1$:     .WORD   40000,100                 ;FSRC OPERAND.
116624  047600  000000          2$:     .WORD   47600,0           ;EXPECTED RESULT.
116630  047600  000001          3$:     .WORD   47600,1           ;ANTICIPATED ERRONEOUS RESULT.
116634  000157                  4$:     157                       ;FPS BEFORE EXECUTION.
116636  000140                          140                       ;FPS AFTER EXECUTION.
116640  177777                          -1                        ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=100000,0 (MOST NEG #)    FL=0
116642  004737  116720          KKC23:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116646  100000  000000          1$:     .WORD   100000,0                  ;FSRC OPERAND.
116652  144000  000000          2$:     .WORD   144000,0                  ;EXPECTED RESULT.
116656  143600  000000          3$:     .WORD   143600,0                  ;ANTICIPATED ERRONEOUS RESULT.
116662  000007                  4$:     7                         ;FPS BEFORE EXECUTION.
116664  000010                          10                        ;FPS AFTER EXECUTION.
116666  177777                          -1                        ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=100000,0          FL=1
116670  004737  116720          KKC24:  JSR     PC,@#LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
116674  100000  000000          1$:     .WORD   100000,0                  ;FSRC OPERAND.
116700  150000  000000          2$:     .WORD   150000,0                  ;EXPECTED RESULT.
116704  147600  000000          3$:     .WORD   147600,0                  ;ANTICIPATED ERRONEOUS RESULT.
116710  000107                  4$:     107                       ;FPS BEFORE EXECUTION.
116712  000110                          110                       ;FPS AFTER EXECUTION.
116714  177777                          -1                        ;ANTICIPATED ERRONEOUS FPS.
116716  000441                  6$:     BR      KKCDONE

                                ;THIS SUBROUTINE, LDCFSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE LDCIF OR LDCLF INSTRUCTION AND CHECK THE RESULTS. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;                       JSR      PC,@#LDCFSUB
                                ;                       ACARG:   .WORD   X,X            ;AC OPERAND
                                ;                       RES:     .WORD   X,X            ;EXPECTED RESULT
                                ;                       ERRES:   .WORD   X,X            ;ERROR RESULT
                                ;                       FPSB:    .WORD   X              ;FPS BEFORE EXECUTION
                                ;                       FPSA:    .WORD   X              ;FPS AFTER EXECUTION
                                ;                       ERFPS:   .WORD   X              ;ERROR FPS
                                ;                       ERR1:    ERROR   X              ;DATA ERROR
                                ;                                BR      CONT
                                ;                       ERR2:    ERROR   X              ;FPS ERROR
                                ;                       CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE LDCIF OR LDCLF INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCFSUB RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCFSUB WILL
                                ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCFSUB WILL RETURN
                                ;TO THE ERROR CALL AT ERR2, OTHERWISE LDCFSUB ITSELF
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;LDCIF OR LDCLF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCFSUB
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCFSUB
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
```

```

116720  012601                          LDCFSUB:           MOV      (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
116722  016100  000014                            MOV      14(R1),R0         ;SET THE FPS.
116726  170100                                    LDFPS    R0
116730  010100                                    MOV      R1,R0

116732  177010                          1$:       LDCIF    (R0),AC0                  ;TEST INSTRUCTION LDCIF OR LDCLF.

116734  170204                                    STFPS    R4                ;GET FPS.
116736  012700  117012                            MOV      #LDCT,R0          ;GET THE RESULT.
116742  012702  000200                            MOV      #200,R2
116746  170102                                    LDFPS    R2
116750  174010                                    STD      AC0,(R0)

116752  012702  117012                            MOV      #LDCT,R2          ;SEE IF THE RESULT WAS CORRECT.
116756  010100                                    MOV      R1,R0
116760  062700  000004                            ADD      #4,R0
116764  012703  000002                            MOV      #2,R3
116770  022022                          2$:       CMP      (R0)+,(R2)+
116772  001006                                    BNE      10$               ;BR IF INCORRECT.
116774  077303                                    SOB      R3,2$

116776  026104  000016                            CMP      16(R1),R4         ;SEE IF THE FPS WAS CORRECT.
117002  001002                                    BNE      10$               ;BR IF INCORRECT.
117004  000161  000022                  3$:       JMP      22(R1)            ;RETURN.
117010                                  10$:
117010  104000                                    EMT                        ;

                                        ;DATA BUFFER:
117012  000000  000000  000000          LDCT:     .WORD    0,0,0,0
117020  000000

117022                                  KKCDONE:
117022  004767  005544                            JSR      PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                                             ;SEE IF THE USER HAS EXPRESSED
                                                                             ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                             ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                             ;THE USER TYPED CONTROL G?).



                                        ;*****************************************************************************
                                        ;TEST 570        LDCID AND LDCLD TEST
                                        ;*****************************************************************************
117026                                  TS570:
                                        ;OPERAND=0         FL=0,    FD=1
117026  004737  117504                  LLC1:     JSR      PC,@#LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
117032  000000  000000                  1$:       .WORD    0,0               ;FSRC OPERAND.
117036  000000  000000  000000          2$:       .WORD    0,0,0,0           ;EXPECTED RESULT.
117044  000000
117046  177777  177777  177777          3$:       .WORD    -1,-1,-1,-1               ;ANTICIPATED ERRONEOUS RESULT.
117054  177777
117056  000213                          4$:       213                        ;FPS BEFORE EXECUTION.
117060  000204                                    204                        ;FPS AFTER EXECUTION.
117062  177777                                    -1                         ;ANTICIPATED ERRONEOUS FPS.
                                        ;OPERAND=0         FL=0,    FD=1
117064  004737  117504                  LLC2:     JSR      PC,@#LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
```

```
117070  000000  177777          1$:     .WORD   0,-1                    ;FSRC OPERAND.
117074  000000  000000  000000  2$:     .WORD   0,0,0,0                 ;EXPECTED RESULT.
117102  000000
117104  004177  177400  000000  3$:     .WORD   4177,177400,0,0                 ;ANTICIPATED ERRONEOUS RESULT.
117112  000000
117114  000200                  4$:     200                             ;FPS BEFORE EXECUTION.
117116  000204                          204                             ;FPS AFTER EXECUTION.
117120  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=0      FL=1    FD=1
117122  004737  117504          LLC3:   JSR     PC,@#LDCDSUB            ;GO EXECUTE THE INSTRUCTION.
117126  000000  000000          1$:     .WORD   0,0                     ;FSRC OPERAND.
117132  000000  000000  000000  2$:     .WORD   0,0,0,0                 ;EXPECTED RESULT.
117140  000000
117142  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1                     ;ANTICIPATED ERRONEOUS RESULT.
117150  177777
117152  000211                  4$:     211                             ;FPS BEFORE EXECUTION.
117154  000204                          204                             ;FPS AFTER EXECUTION.
117156  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=40000  FL=0    FD=1
117160  004737  117504          LLC4:   JSR     PC,@#LDCDSUB            ;GO EXECUTE THE INSTRUCTION.
117164  040000  000000          1$:     .WORD   40000,0                 ;FSRC OPERAND.
117170  043600  000000  000000  2$:     .WORD   43600,0,0,0                     ;EXPECTED RESULT.
117176  000000
117200  047600  000000  000000  3$:     .WORD   47600,0,0,0                     ;ANTICIPATED ERRONEOUS RESULT.
117206  000000
117210  000217                  4$:     217                             ;FPS BEFORE EXECUTION.
117212  000200                          200                             ;FPS AFTER EXECUTION.
117214  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=-40000 FL=0    FD=1
117216  004737  117504          LLC5:   JSR     PC,@#LDCDSUB            ;GO EXECUTE THE INSTRUCTION.
117222  140000  000000          1$:     .WORD   -40000,0                        ;FSRC OPERAND.
117226  143600  000000  000000  2$:     .WORD   143600,0,0,0                    ;EXPECTED RESULT.
117234  000000
117236  043600  000000  000000  3$:     .WORD   43600,0,0,0                     ;ANTICIPATED ERRONEOUS RESULT.
117244  000000
117246  000200                  4$:     200                             ;FPS BEFORE EXECUTION.
117250  000210                          210                             ;FPS AFTER EXECUTION.
117252  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=40000,0        FL=1    FD=1
117254  004737  117504          LLC6:   JSR     PC,@#LDCDSUB            ;GO EXECUTE THE INSTRUCTION.
117260  040000  000000          1$:     .WORD   40000,0                 ;FSRC OPERAND.
117264  047600  000000  000000  2$:     .WORD   47600,0,0,0                     ;EXPECTED RESULT.
117272  000000
117274  043600  000000  000000  3$:     .WORD   43600,0,0,0                     ;ANTICIPATED ERRONEOUS RESULT.
117302  000000
117304  000317                          317                             ;FPS BEFORE EXECUTION.
117306  000300                          300                             ;FPS AFTER EXECUTION.
117310  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=0,1    FL=1    FD=1
117312  004737  117504          LLC7:   JSR     PC,@#LDCDSUB            ;GO EXECUTE THE INSTRUCTION.
117316  000000  000001          1$:     .WORD   0,1                     ;FSRC OPERAND.
117322  040200  000000  000000  2$:     .WORD   40200,0,0,0                     ;EXPECTED RESULT.
117330  000000
117332  034200  000000  000000  3$:     .WORD   34200,0,0,0                     ;ANTICIPATED ERRONEOUS RESULT.
117340  000000
117342  000300                  4$:     300                             ;FPS BEFORE EXECUTION.
```

```
117344  000300                                  300                             ;FPS AFTER EXECUTION.
117346  177777                                  -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=77777,177777   FL=1    FD=1
117350  004737  117504          LLC10:  JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
117354  077777  177777          1$:     .WORD   77777,177777            ;FSRC OPERAND.
117360  047777  177777  177000  2$:     .WORD   47777,177777,177000,0           ;EXPECTED RESULT.
117366  000000
117370  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1                     ;ANTICIPATED ERRONEOUS RESULT.
117376  177777
117400  000317                  4$:     317                             ;FPS BEFORE EXECUTION.
117402  000300                          300                             ;FPS AFTER EXECUTION.
117404  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=-PATTERN       FL=1    FD=1

117406  004767  000072          LLC11:  JSR     PC,LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
117412  177777  177526          1$:     .WORD   -1,-252         ;FSRC OPERAND.
117416  142052  000000  000000  2$:     .WORD   142052,0,0,0            ;EXPECTED RESULT.
117424  000000
117426  136052  000000  000000  3$:     .WORD   136052,0,0,0            ;ANTICIPATED ERRONEOUS RESULT.
117434  000000
117436  000307                  4$:     307                             ;FPS BEFORE EXECUTION.
117440  000310                          310                             ;FPS AFTER EXECUTION.
117442  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
                                ;OPERAND=PATTERN        FL=1    FD=1    FT=1
117444  004767  000034          LLC12:  JSR     PC,LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
117450  012345  067012          1$:     .WORD   12345,67012             ;FSRC OPERAND.
117454  047247  025560  050000  2$:     .WORD   47247,025560,050000,0           ;EXPECTED RESULT.
117462  000000
117464  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1                     ;ANTICIPATED ERRONEOUS RESULT.
117472  177777
117474  000352                  4$:     352                             ;FPS BEFORE EXECUTION.
117476  000340                          340                             ;FPS AFTER EXECUTION.
117500  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
117502  000435                  6$:     BR      LLCDONE

                                ;THIS SUBROUTINE, LDCDSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE LDCID OR LDCLD INSTRUCTION AND CHECK THE RESULTS. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               JSR     PC,@#LDCDSUB
                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
                                ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
                                ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               ERFPS:  .WORD   X               ;ERROR FPS.
                                ;               ERR1:   ERROR   X               ;DATA ERROR.
                                ;                       BR      CONT
                                ;               ERR2:   ERROR   X               ;FPS ERROR.
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE LDCID OR LDCLD INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCDSUB RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCDSUB
```

```
                                        ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCDSUB WILL RETURN
                                        ;TO THE ERROR CALL AT ERR2, OTHERWISE LDCDSUB ITSELF
                                        ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                        ;LDCID OR LDCLD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                        ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                        ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCDSUB
                                        ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                        ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCDSUB WILL
                                        ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

117504  012601                          LDCDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
117506  016100  000024                          MOV     24(R1),R0       ;SET THE FPS.
117512  170100                                  LDFPS   R0
117514  010100                                  MOV     R1,R0
117516  177010                          1$:     LDCID   (R0),AC0        ;TEST INSTRUCTION, LDCID OR LDCLD.

117520  170204                                  STFPS   R4              ;GET FPS.
117522  012700  117012                          MOV     #LDCT,R0        ;GET THE RESULT.
117526  012702  000200                          MOV     #200,R2
117532  170102                                  LDFPS   R2
117534  174010                                  STD     AC0,(R0)

                                        ;SEE IF THE RESULT IS CORRECT.
117536  012702  117012                          MOV     #LDCT,R2
117542  010100                                  MOV     R1,R0
117544  062700  000004                          ADD     #4,R0
117550  012703  000002                          MOV     #2,R3
117554  022022                          2$:     CMP     (R0)+,(R2)+
117556  001006                                  BNE     10$             ;BR IF INCORRECT.
117560  077303                                  SOB     R3,2$

117562  026104  000026                          CMP     26(R1),R4               ;IS THE FPS CORRECT?
117566  001002                                  BNE     10$             ;BR IF INCORRECT.
117570  000161  000032                  3$:     JMP     32(R1)          ;RETURN.
117574                                  10$:
117574  104000                                  EMT                     ;

117576                                  LLCDONE:
117576  004767  004770                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                        ;SEE IF THE USER HAS EXPRESSED
                                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                        ;THE USER TYPED CONTROL G?).


                                        ;*****************************************************************************
                                        ;TEST 571       LDEXP TEST
                                        ;*****************************************************************************
117602                                  TS571:

                                        ; NON-ZERO RES. VALID EXPON=210 (EXCESS 200)=10
117602  004767  001136                  MMC1:   JSR     PC,LDXSUB       ;GO EXECUTE THE INSTRUCTION.
117606  012345  067012  034567          1$:     .WORD   12345,67012,34567,012345        ;AC0 OPERAND.
117614  012345
117616  000010                          2$:     .WORD   10              ;EXPONENT OPERAND.
117620  042145  067012  034567          3$:     .WORD   42145,67012,34567,012345                ;EXPECTED RESULT.
```

```
117626  012345
117630  002145  067012  034567  4$:     .WORD   2145,67012,34567,012345         ;ANTICIPATED ERRONEOUS RESULT.
117636  012345
117640  047217                  5$:     47217                           ;FPS BEFORE EXECUTION.
117642  047200                          47200                           ;FPS AFTER EXECUTION.
117644  147200                          147200                          ;ANTICIPATED ERRONEOUS FPS.
117646  177777                          -1                      ;EXPECTED FEC.
                                ;NON-ZERO       RES     NEG.
117650  004737  120744          MMC2:   JSR     PC,@#LDXSUB     ;EXPON=377
117654  123456  070123  045670  1$:     .WORD   123456,70123,45670,123456       ;AC0 OPERAND.
117662  123456
117664  000177                  2$:     .WORD   177                     ;EXPONENT OPERAND.
117666  177656  070123  045670  3$:     .WORD   177656,70123,45670,123456               ;EXPECTED RESULT.
117674  123456
117676  137656  070123  045670  4$:     .WORD   137656,70123,45670,123456               ;ANTICIPATED ERRONEOUS RESULT.
117704  123456
117706  047207                  5$:     47207                           ;FPS BEFORE EXECUTION.
117710  047210                          47210                           ;FPS AFTER EXECUTION.
117712  147210                          147210                          ;ANTICIPATED ERRONEOUS FPS.
117714  177777                          -1                      ;EXPECTED FEC.
                                ;NON-ZERO       RES,    EXP=256=(56)REAL
117716  004737  120744          MMC3:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
117722  073261  057645  043323  1$:     .WORD   73261,057645,43323,101760       ;AC0 OPERAND.
117730  101760
117732  000056                  2$:     .WORD   56                      ;EXPONENT OPERAND.
117734  053461  057645  043323  3$:     .WORD   53461,057645,43323,101760               ;EXPECTED RESULT.
117742  101760
117744  177777  177777  177777  4$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
117752  177777
117754  047200                  5$:     47200                           ;FPS BEFORE EXECUTION.
117756  047200                          47200                           ;FPS AFTER EXECUTION.
117760  147200                          147200                          ;ANTICIPATED ERRONEOUS FPS.
117762  177777                          -1                      ;EXPECTED FEC.
                                ;EXP=27 (EXCESS 200)=-151 (OCT)
117764  004737  120744          MMC4:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
117770  012223  024252  062720  1$:     .WORD   12223,24252,62720,21222 ;AC0 OPERAND.
117776  021222
120000  177627                  2$:     .WORD   -151                    ;EXPONENT OPERAND.
120002  005623  024252  062720  3$:     .WORD   5623,24252,62720,21222          ;EXPECTED RESULT.
120010  021222
120012  177777  177777  177777  4$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
120020  177777
120022  047200                  5$:     47200                           ;FPS BEFORE EXECUTION.
120024  047200                          47200                           ;FPS AFTER EXECUTION.
120026  147200                          147200                          ;ANTICIPATED ERRONEOUS FPS.
120030  177777                          -1                      ;EXPECTED FEC.
                                ;EXP=0 (EXCESS 200)=-200 (OCT), POSITIVE FRAC
                                ; FIV=1
120032  004737  120744          MMC5:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120036  030131  032334  035363  1$:     .WORD   30131,32334,35363,73031 ;AC0 OPERAND.
120044  073031
120046  177600                  2$:     .WORD   -200                    ;EXPONENT OPERAND.
120050  000131  032334  035363  3$:     .WORD   00131,32334,35363,73031         ;EXPECTED RESULT.
120056  073031
120060  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
120066  000000
```

```
120070  042200                          5$:     42200                           ;FPS BEFORE EXECUTION.
120072  142204                                  142204                          ;FPS AFTER EXECUTION.
120074  042202                                  42202                           ;ANTICIPATED ERRONEOUS FPS.
120076  000012                                  12                      ;EXPECTED FEC.
                                        ;EXP=0  (EXCESS 200)=-200 (OCT), NEG FRACT,FIU=1
120100  004737  120744                  MMC6:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120104  140414  024344  045464          1$:     .WORD   140414,24344,45464,74045        ;AC0 OPERAND.
120112  074045
120114  177600                          2$:     .WORD   -200                    ;EXPONENT OPERAND.
120116  100014  024344  045464          3$:     .WORD   100014,24344,45464,74045        ;-0             ;EXPECTED RESULT.
120124  074045
120126  000000  000000  000000          4$:     .WORD   0,0,0,0                 ;ANTICIPATED ERRONEOUS RESULT.
120134  000000
120136  042200                          5$:     42200                           ;FPS BEFORE EXECUTION.
120140  142214                                  142214                          ;FPS AFTER EXECUTION.
120142  042214                                  42214                           ;ANTICIPATED ERRONEOUS FPS.
120144  000012                                  12                      ;EXPECTED FEC.
                                        ;EXP=0 (EXCESS 200)=-200 (OCT),POS FRAC, FIU=0

120146  004737  120744                  MMC7:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120152  051525  035455  005675          1$:     .WORD   51525,35455,5675,05152  ;AC0 OPERAND.
120160  005152
120162  177600                          2$:     .WORD   -200                    ;EXPONENT OPERAND.
120164  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;EXPECTED RESULT.
120172  000000
120174  000125  035455  005675          4$:     .WORD   00125,35455,5675,05152          ;ANTICIPATED ERRONEOUS RESULT.
120202  005152
120204  045200                                  45200                           ;FPS BEFORE EXECUTION.
120206  045204                                  45204                           ;FPS AFTER EXECUTION.
120210  145204                                  145204                          ;ANTICIPATED ERRONEOUS FPS.
120212  177777                                  -1                      ;EXPECTED FEC.
                                        ;EXP=-1405 (EXCESS 200)=-1605 (OCT), FIU=1
120214  004737  120744                  MMC10:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120220  061626  062636  046566          1$:     .WORD   61626,62636,46566,67606 ;AC0 OPERAND.
120226  067606
120230  176173                          2$:     .WORD   -1605                   ;EXPONENT OPERAND.
120232  076626  062636  046566          3$:     .WORD   76626,62636,46566,67606         ;EXPECTED RESULT.
120240  067606
120242  000000  000000  000000          4$:     .WORD   0,0,0,0                 ;ANTICIPATED ERRONEOUS RESULT.
120250  000000
120252  042200                          5$:     42200                           ;FPS BEFORE EXECUTION.
120254  142200                                  142200                          ;FPS AFTER EXECUTION.
120256  042204                                  42204                           ;ANTICIPATED ERRONEOUS FPS.
120260  000012                                  12                      ;EXPECTED FEC.
                                        ;EXP=-17416 (EXCESS 200)=-17616 (OCT), FIU=0
120262  004737  120744                  MMC11:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120266  071727  037475  076777          1$:     .WORD   71727,37475,76777,17273 ;AC0 OPERAND.
120274  017273
120276  160162                          2$:     .WORD   -17616                  ;EXPONENT OPERAND.
120300  000000  000000  000000          3$:     .WORD   0,0,0,0                 ;EXPECTED RESULT.
120306  000000
120310  074527  037475  076777          4$:     .WORD   74527,37475,76777,17273         ;ANTICIPATED ERRONEOUS RESULT.
120316  017273
120320  045200                          5$:     45200                           ;FPS BEFORE EXECUTION.
120322  045204                                  45204                           ;FPS AFTER EXECUTION.
120324  145200                                  145200                          ;ANTICIPATED ERRONEOUS FPS.
```

```
120326  177777                                -1                    ;EXPECTED FEC.
                                      ;EXP=-1601 (EXCESS 200)=-2001 (OCT), FIU=1
120330  004737  120744                MMC12:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120334  001020  030405  006070        1$:     .WORD   01020,30405,06070,00102 ;ACO OPERAND.
120342  000102
120344  175777                        2$:     .WORD   -2001           ;EXPONENT OPERAND.
120346  037620  030405  006070        3$:     .WORD   37620,30405,06070,00102         ;EXPECTED RESULT.
120354  000102
120356  000000  000000  000000        4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
120364  000000
120366  042200                        5$:     42200                   ;FPS BEFORE EXECUTION.
120370  142200                                142200                  ;FPS AFTER EXECUTION.
120372  042204                                42204                   ;ANTICIPATED ERRONEOUS FPS.
120374  000012                                12                    ;EXPECTED FEC.
                                      ;EXP=1206 (EXCESS 200)=1006 (OCT) FIV =1
120376  004737  120744                MMC13:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120402  012131  014151  016171        1$:     .WORD   12131,14151,16171,10111 ;ACO OPERAND.
120410  010111
120412  001006                        2$:     .WORD   1006            ;EXPONENT OPERAND.
120414  041531  014151  016171        3$:     .WORD   41531,14151,16171,10111         ;EXPECTED RESULT.
120422  010111
120424  000000  000000  000000        4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
120432  000000
120434  041200                        5$:     41200                   ;FPS BEFORE EXECUTION.
120436  141202                                141202                  ;FPS AFTER EXECUTION.
120440  041204                                41204                   ;ANTICIPATED ERRONEOUS FPS.
120442  000010                                10                    ;EXPECTED FEC.
                                      ;EXP=16315 (EXCESS 200)=16115 (OCT) FIV=0
120444  004737  120744                MMC14:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120450  027262  025242  023222        1$:     .WORD   27262,25242,23222,21202 ;ACO OPERAND.
120456  021202
120460  016115                        2$:     .WORD   16115           ;EXPONENT OPERAND.
120462  000000  000000  000000        3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
120470  000000
120472  063262  025242  023222        4$:     .WORD   63262,25242,23222,21202         ;ANTICIPATED ERRONEOUS RESULT.
120500  021202
120502  046200                        5$:     46200                   ;FPS BEFORE EXECUTION.
120504  046206                                46206                   ;FPS AFTER EXECUTION.
120506  146202                                146202                  ;ANTICIPATED ERRONEOUS FPS.
120510  177777                                -1                    ;EXPECTED FEC.
                                      ;EXP=11011 (EXCESS 200)=10611 (OCT) FIV=1

120512  004737  120744                MMC15:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
120516  030313  032333  034353        1$:     .WORD   30313,32333,34353,36373 ;ACO OPERAND.
120524  036373
120526  010611                        2$:     .WORD   10611           ;EXPONENT OPERAND.
120530  002313  032333  034353        3$:     .WORD   2313,32333,34353,36373          ;EXPECTED RESULT.
120536  036373
120540  000000  000000  000000        4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
120546  000000
120550  041200                        5$:     41200                   ;FPS BEFORE EXECUTION.
120552  141202                                141202                  ;FPS AFTER EXECUTION.
120554  041204                                41204                   ;ANTICIPATED ERRONEOUS FPS.
120556  000010                                10                    ;EXPECTED FEC.
                                      ;EXP=17123 (EXCESS 200)=16723 (OCT) FIV=0
```

```
120560  004737  120744           MMC16:  JSR     PC,@#LDXSUB         ;GO EXECUTE THE INSTRUCTION.
120564  040414  042434  044454   1$:     .WORD   40414,42434,44454,46474 ;AC0 OPERAND.
120572  046474
120574  016723                   2$:     .WORD   16723               ;EXPONENT OPERAND.
120576  000000  000000  000000   3$:     .WORD   0,0,0,0             ;EXPECTED RESULT.
120604  000000
120606  024614  042434  044454   4$:     .WORD   24614,42434,44454,46474        ;ANTICIPATED ERRONEOUS RESULT.
120614  046474
120616  046200                   5$:     46200                       ;FPS BEFORE EXECUTION.
120620  046206                           46206                       ;FPS AFTER EXECUTION.
120622  146202                           146202                      ;ANTICIPATED ERRONEOUS FPS.
120624  177777                           -1                  ;EXPECTED FEC.
                                 ;EXP=   254 (OCT)=       454 (EXCESS 200)         FIV=1

120626  004737  120744           MMC17:  JSR     PC,@#LDXSUB         ;GO EXECUTE THE INSTRUCTION.
120632  050515  052535  054555   1$:     .WORD   50515,52535,54555,56575 ;AC0 OPERAND.
120640  056575
120642  000254                   2$:     .WORD   254                 ;EXPONENT OPERAND.
120644  013115  052535  054555   3$:     .WORD   13115,52535,54555,56575        ;EXPECTED RESULT.
120652  056575
120654  000000  000000  000000   4$:     .WORD   0,0,0,0             ;ANTICIPATED ERRONEOUS RESULT.
120662  000000
120664  041200                   5$:     41200                       ;FPS BEFORE EXECUTION.
120666  141202                           141202                      ;FPS AFTER EXECUTION.
120670  041204                           41204                       ;ANTICIPATED ERRONEOUS FPS.
120672  000010                           10                  ;EXPECTED FEC.
                                 ;EXP=   313 (OCT)=       513(EXCESS 200) FIV=0

120674  004737  120744           MMC20:  JSR     PC,@#LDXSUB         ;GO EXECUTE THE INSTRUCTION.
120700  060616  062636  064656   1$:     .WORD   60616,62636,64656,66676 ;AC0 OPERAND.
120706  066676
120710  000313                   2$:     .WORD   313                 ;EXPONENT OPERAND.
120712  000000  000000  000000   3$:     .WORD   0,0,0,0             ;EXPECTED RESULT.
120720  000000
120722  022616  062636  064656   4$:     .WORD   22616,62636,64656,66676        ;ANTICIPATED ERRONEOUS RESULT.
120730  066676
120732  046200                   5$:     46200                       ;FPS BEFORE EXECUTION.
120734  046206                           46206                       ;FPS AFTER EXECUTION.
120736  146202                           146202                      ;ANTICIPATED ERRONEOUS FPS.
120740  177777                           -1                  ;EXPECTED FEC.
120742  000457                           BR      MMCDONE

                                 ;THIS SUBROUTINE, LDXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
                                 ;THE LDEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
                                 ;TO IT IS MADE THUS:
                                 ;
                                 ;                JSR     PC,@#LDXSUB
                                 ;                ACARG:  .WORD   X,X,X,X           ;AC OPERAND
                                 ;                EXP:    .WORD   X                 ;EXPONENT
                                 ;                RES:    .WORD   X,X,X,X           ;EXPECTED RESULT
                                 ;                ERRES:  .WORD   X,X,X,X           ;ERROR RESULT
                                 ;                FPSB:   .WORD   X                 ;FPS BEFORE EXECUTION
                                 ;                FPSA:   .WORD   X                 ;FPS AFTER EXECUTION
                                 ;                ERFPS:  .WORD   X                 ;ERROR FPS.
                                 ;                FEC:    .WORD   X                 ;EXPECTED FEC
                                 ;                ERR1:   ERROR   X                 ;DATA ERROR.
```

```
                                ;                       BR      CONT
                                ;               ERR2:   ERROR   X                       ;FPS ERROR.
                                ;               CONT:                                   ;RETURN ADDRESS

                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE LDEXP INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDXSUB RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDXSUB
                                ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDXSUB WILL RETURN
                                ;TO THE ERROR CALL AT ERR2, OTHERWISE LDXSUB ITSELF
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;LDEXP IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDXSUB
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDXSUB WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

120744  012601                  LDXSUB: MOV     (SP)+,R1                ;GET A POINTER TO THE ARGUMENTS.
120746  012700  000200                  MOV     #200,R0                 ;LOAD THE AC0 OPERAND.
120752  170100                          LDFPS   R0
120754  010100                          MOV     R1,R0
120756  172410                          LDD     (R0),AC0
120760  016100  000032                  MOV     32(R1),R0               ;SET UP THE FPS.
120764  170100                          LDFPS   R0
120766  010100                          MOV     R1,R0
120770  062700  000010                  ADD     #10,R0

120774  176410                  1$:     LDEXP   (R0),AC0                ;TEST INSTRUCTION.

120776  170204                          STFPS   R4                      ;GET THE FPS.
121000  170305                          STST    R5                      ;GET THE FEC.
121002  012700  000200                  MOV     #200,R0                 ;GET THE RESULT.
121006  170100                          LDFPS   R0
121010  012700  121072                  MOV     #LDXT,R0
121014  174010                          STD     AC0,(R0)
121016  012702  121072                  MOV     #LDXT,R2                ;SEE IF THE RESULT WAS CORRECT.
121022  010103                          MOV     R1,R3
121024  062703  000012                  ADD     #12,R3
121030  012700  000004                  MOV     #4,R0
121034  022223                  2$:     CMP     (R2)+,(R3)+
121036  001014                          BNE     10$                     ;BRANCH IF NOT CORRECT.
121040  077003                          SOB     R0,2$
121042  020461  000034                  CMP     R4,34(R1)               ;SEE IF THE FPS WAS CORRECT.
121046  001010                          BNE     10$                     ;BRANCH IF NOT CORRECT.
121050  005761  000034                  TST     34(R1)
121054  100003                          BPL     3$
121056  020561  000040                  CMP     R5,40(R1)               ;SEE IF THE FEC WAS CORRECT.
121062  001002                          BNE     10$                     ;BRANCH IF NOT CORRECT.

121064  000161  000042          3$:     JMP     42(R1)                  ;RETURN.
121070                          10$:
121070  104000                          EMT                             ;

                                ;DATA BUFFER:
```

```
121072  000000  000000  000000  LDXT:   .WORD   0,0,0,0
121100  000000

121102                          MMCDONE:
121102  004767  003464                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).



                                ;*************************************************************************
                                ;TEST 572       DESTINATION MODES, MODE 1 (FL=0), TEST
                                ;*************************************************************************
121106                          TS572:


121106  012700  121176                  MOV     #NNCTB0,R0      ;SET UP THE DATA BUFFER.
121112  012701  000006                  MOV     #6,R1
121116  012720  177777          1$:     MOV     #-1,(R0)+
121122  077103                          SOB     R1,1$
121124  012700  102345                  MOV     #102345,R0
121130  012737  121212  000004          MOV     #NNC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
121136  170100                          LDFPS   R0              ;SET UP FPS.
121140  012700  121202                  MOV     #NNCTB1,R0

121144  170210                  NNC2:   STFPS   (R0)            ;TEST INSTRUCTION.
121146  020027  121202                  CMP     R0,#NNCTB1      ;IS R0 CORRECT?
121152  001017                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
121154  023727  121202  102345          CMP     @#NNCTB1,#102345   ;IS RESULT CORRECT?
121162  001013                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
121164  023727  121204  177777          CMP     @#NNCTB1+2,#-1  ;IS THE RESULT CORRECT?
121172  001007                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
121174  000407                          BR      NNCDONE

                                ;TEST DATA BUFFER:
121176  177777  177777          NNCTB0: .WORD   -1,-1
121202  177777  177777  177777  NNCTB1: .WORD   -1,-1,-1,-1
121210  177777
121212                          NNC10:
121212  104000                          EMT                     ;

121214                          NNCDONE:
121214  004767  003352                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*************************************************************************
                                ;TEST 573       DESTINATION MODES, MODE 2 (FL=0), TEST
                                ;*************************************************************************
121220                          TS573:
```

```

121220  012700  121310                          MOV     #OOCTB0,R0          ;SET UP THE DATA BUFFER.
121224  012701  000006                          MOV     #6,R1
121230  012720  177777                  1$:     MOV     #-1,(R0)+
121234  077103                                  SOB     R1,1$
121236  012700  105412                          MOV     #105412,R0
121242  012737  121324  000004                  MOV     #OOC10,@#ERRVECT    ;SET UP FOR TRAPS TO VECTOR 4.
121250  170100                                  LDFPS   R0                  ;SET UP FPS.
121252  012700  121314                          MOV     #OOCTB1,R0

121256  170220                          OOC2:   STFPS   (R0)+               ;TEST INSTRUCTION.
121260  020027  121316                          CMP     R0,#OOCTB1+2        ;IS R0 CORRECT?
121264  001017                                  BNE     OOC10               ;BRANCH IF NOT CORRECT.
121266  023727  121314  105412                  CMP     @#OOCTB1,#105412    ;IS THE RESULT CORRECT?
121274  001013                                  BNE     OOC10               ;BRANCH IF NOT CORRECT.
121276  023727  121316  177777                  CMP     @#OOCTB1+2,#-1      ;IS THE RESULT CORRECT?
121304  001007                                  BNE     OOC10               ;BRANCH IF NOT CORRECT.
121306  000407                                  BR      OOCDONE


                                        ;TEST DATA BUFFER:
121310  177777  177777                  OOCTB0: .WORD   -1,-1
121314  177777  177777  177777          OOCTB1: .WORD   -1,-1,-1,-1
121322  177777
121324                                  OOC10:
121324  104000                                  EMT                         ;

121326                                  OOCDONE:
121326  004767  003240                          JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                            ;SEE IF THE USER HAS EXPRESSED
                                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                            ;THE USER TYPED CONTROL G?).



                                        ;*****************************************************************************
                                        ;TEST 574       DESTINATION MODES, MODE 4 (FL=0), TEST
                                        ;*****************************************************************************
121332                                  TS574:

121332  012700  121422                          MOV     #PPCTB0,R0          ;SET UP THE DATA BUFFER.
121336  012701  000006                          MOV     #6,R1
121342  012720  177777                  1$:     MOV     #-1,(R0)+
121346  077103                                  SOB     R1,1$
121350  012700  105555                          MOV     #105555,R0
121354  012737  121436  000004                  MOV     #PPC10,@#ERRVECT    ;SET UP FOR TRAPS TO VECTOR 4.
121362  170100                                  LDFPS   R0                  ;SET UP FPS.
121364  012700  121430                          MOV     #PPCTB1+2,R0

121370  170240                          PPC2:   STFPS   -(R0)               ;TEST INSTRUCTION.
121372  020027  121426                          CMP     R0,#PPCTB1          ;IS R0 CORRECT?
121376  001017                                  BNE     PPC10               ;BRANCH IF NOT CORRECT.
121400  023727  121426  105555                  CMP     @#PPCTB1,#105555    ;IS THE RESULT CORRECT?
121406  001013                                  BNE     PPC10               ;BRANCH IF NOT CORRECT.
121410  023727  121430  177777                  CMP     @#PPCTB1+2,#-1      ;IS THE RESULT CORRECT?
121416  001007                                  BNE     PPC10               ;BRANCH IF NOT CORRECT.
```

```
121420  000407                           BR      PPCDONE

                                 ;TEST DATA BUFFER:
121422  177777  177777           PPCTB0: .WORD   -1,-1
121426  177777  177777  177777   PPCTB1: .WORD   -1,-1,-1,-1
121434  177777
121436                           PPC10:
121436  104000                           EMT                     ;
121440                           PPCDONE:
121440  004767  003126                   JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).

                                 ;*******************************************************************************
                                 ;TEST 575        DESTINATION MODES, MODE 3 (FL=0), TEST
                                 ;*******************************************************************************
121444                           TS575:

121444  012700  121540                   MOV     #QQCTB0,R0      ;SET UP THE DATA BUFFER.
121450  012701  000010                   MOV     #10,R1
121454  012720  177777           1$:     MOV     #-1,(R0)+
121460  077103                           SOB     R1,1$
121462  012700  106653                   MOV     #106653,R0
121466  012737  121560  000004           MOV     #QQC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
121474  170100                           LDFPS   R0              ;SET UP FPS.
121476  012700  121554                   MOV     #QQCTB2,R0
121502  012710  121544                   MOV     #QQCTB1,(R0)

121506  170230                   QQC2:   STFPS   @(R0)+          ;TEST INSTRUCTION.
121510  020027  121556                   CMP     R0,#QQCTB2+2    ;IS R0 CORRECT?
121514  001021                           BNE     QQC10           ;BRANCH IF NOT CORRECT.
121516  023727  121544  106653           CMP     @#QQCTB1,#106653  ;IS THE RESULT CORRECT?
121524  001015                           BNE     QQC10           ;BRANCH IF NOT CORRECT.
121526  023727  121554  121544           CMP     @#QQCTB2,#QQCTB1  ;IS THE RESULT CORRECT?
121534  001011                           BNE     QQC10           ;BRANCH IF NOT CORRECT.
121536  000411                           BR      QQCDONE

                                 ;TEST DATA BUFFER:
121540  177777  177777           QQCTB0: .WORD   -1,-1
121544  177777  177777  177777   QQCTB1: .WORD   -1,-1,-1,-1
121552  177777
121554  177777  177777           QQCTB2: .WORD   -1,-1
121560                           QQC10:
121560  104000                           EMT                     ;
121562                           QQCDONE:
121562  004767  003004                   JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                 ;SEE IF THE USER HAS EXPRESSED
                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                 ;THE USER TYPED CONTROL G?).
```

```

                                          ;*******************************************************************************
                                          ;TEST 576          DESTINATION MODES, MODE 5 (FL=0), TEST
                                          ;*******************************************************************************
 121566                                   TS576:


 121566  012700  121664                           MOV     #RRCTB0,R0          ;SET UP THE DATA BUFFER.
 121572  012701  000006                           MOV     #6,R1
 121576  012720  177777                   1$:     MOV     #-1,(R0)+
 121602  077103                                   SOB     R1,1$
 121604  012700  004301                           MOV     #004301,R0
 121610  012737  121704  000004                   MOV     #RRC10,@#ERRVECT   ;SET UP FOR TRAPS TO VECTOR 4.
 121616  170100                                   LDFPS   R0                 ;SET UP FPS.
 121620  012700  121702                           MOV     #RRCTB2+2,R0
 121624  012760  121670  177776                   MOV     #RRCTB1,-2(R0)

 121632  170250                           RRC2:   STFPS   @-(R0)              ;TEST INSTRUCTION.
 121634  020027  121700                           CMP     R0,#RRCTB2          ;IS R0 CORRECT?
 121640  001021                                   BNE     RRC10               ;BRANCH IF NOT CORRECT.
 121642  023727  121670  004301                   CMP     @#RRCTB1,#004301   ;IS THE RESULT CORRECT?
 121650  001015                                   BNE     RRC10               ;BRANCH IF NOT CORRECT.
 121652  023727  121700  121670                   CMP     @#RRCTB2,#RRCTB1   ;IS THE RESULT CORRECT?
 121660  001011                                   BNE     RRC10               ;BRANCH IF NOT CORRECT.
 121662  000411                                   BR      RRCDONE

                                          ;TEST DATA BUFFER:
 121664  177777  177777                   RRCTB0: .WORD   -1,-1
 121670  177777  177777  177777           RRCTB1: .WORD   -1,-1,-1,-1
 121676  177777
 121700  177777  177777                   RRCTB2: .WORD   -1,-1
 121704                                   RRC10:
 121704  104000                                   EMT                         ;
 121706                                   RRCDONE:
 121706  004767  002660                           JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                              ;SEE IF THE USER HAS EXPRESSED
                                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                              ;THE USER TYPED CONTROL G?).



                                          ;*******************************************************************************
                                          ;TEST 577          DESTINATION MODES, MODE 6 (FL=0), TEST
                                          ;*******************************************************************************
 121712                                   TS577:


 121712  012700  122014                           MOV     #SSCTB0,R0          ;SET UP THE DATA BUFFER.
 121716  012701  000006                           MOV     #6,R1
 121722  012720  177777                   1$:     MOV     #-1,(R0)+
 121726  077103                                   SOB     R1,1$
 121730  012700  102514                           MOV     #102514,R0
 121734  012737  122030  000004                   MOV     #SSC10,@#ERRVECT   ;SET UP FOR TRAPS TO VECTOR 4.
 121742  170100                                   LDFPS   R0                 ;SET UP FPS.
 121744  005001                                   CLR     R1
 121746  012700  114617                           MOV     #SSCTB1-5201,R0
```

```

121752  170260  005201          SSC2:    STFPS   5201(R0)              ;TEST INSTRUCTION.
121756  020127  000000                   CMP     R1,#0                 ;WAS PC CORRECT AFTER EXECUTION?
121762  001022                           BNE     SSC10                 ;BRANCH IF NOT CORRECT.
121764  020027  114617                   CMP     R0,#SSCTB1-5201       ;IS R0 CORRECT?
121770  001017                           BNE     SSC10                 ;BRANCH IF NOT CORRECT.
121772  023727  122020  102514           CMP     @#SSCTB1,#102514      ;IS THE RESULT CORRECT?
122000  001013                           BNE     SSC10                 ;BRANCH IF NOT CORRECT.
122002  023727  122022  177777           CMP     @#SSCTB1+2,#-1        ;IS THE RESULT CORRECT?
122010  001007                           BNE     SSC10                 ;BRANCH IF NOT CORRECT.
122012  000407                           BR      SSCDONE


                                ;TEST DATA BUFFER:
122014  177777  177777          SSCTB0:  .WORD   -1,-1
122020  177777  177777  177777  SSCTB1:  .WORD   -1,-1,-1,-1
122026  177777
122030                          SSC10:
122030  104000                           EMT                           ;
122032                          SSCDONE:
122032  004767  002534                   JSR     PC,.RSET              ;GO INITIALIZE THE FPS AND STACK; AND
                                                                       ;SEE IF THE USER HAS EXPRESSED
                                                                       ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                       ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                       ;THE USER TYPED CONTROL G?).
```

```


                                        ;*******************************************************************************
                                        ;TEST 600          DESTINATION MODES, MODE 7 (FL=0), TEST
                                        ;*******************************************************************************
 122036                                 TS600:

 122036  012700  122146                         MOV     #TTCTB0,R0          ;SET UP THE DATA BUFFER.
 122042  012701  000010                         MOV     #10,R1
 122046  012720  177777                 1$:     MOV     #-1,(R0)+
 122052  077103                                 SOB     R1,1$
 122054  012700  103747                         MOV     #103747,R0
 122060  012737  122166  000004                 MOV     #TTC10,@#ERRVECT    ;SET UP FOR TRAPS TO VECTOR 4.
 122066  170100                                 LDFPS   R0                  ;SET UP FPS.
 122070  005001                                 CLR     R1
 122072  012700  114761                         MOV     #TTCTB2-5201,R0
 122076  012760  122152  005201                 MOV     #TTCTB1,5201(R0)

 122104  170270  005201                 TTC2:   STFPS   @5201(R0)                  ;TEST INSTRUCTION.
 122110  022701  000000                         CMP     #0,R1               ;WAS PC CORRECT AFTER EXECUTION?
 122114  001024                                 BNE     TTC10               ;BRANCH IF NOT CORRECT.
 122116  020027  114761                         CMP     R0,#TTCTB2-5201     ;IS R0 CORRECT?
 122122  001021                                 BNE     TTC10               ;BRANCH IF NOT CORRECT.
 122124  023727  122152  103747                 CMP     @#TTCTB1,#103747    ;IS THE RESULT CORRECT?
 122132  001015                                 BNE     TTC10               ;BRANCH IF NOT CORRECT.
 122134  023727  122154  177777                 CMP     @#TTCTB1+2,#-1      ;IS THE RESULT CORRECT?
 122142  001011                                 BNE     TTC10               ;BRANCH IF NOT CORRECT.
 122144  000411                                 BR      TTCDONE

                                        ;TEST DATA BUFFER:
 122146  177777  177777                 TTCTB0: .WORD   -1,-1
 122152  177777  177777  177777         TTCTB1: .WORD   -1,-1,-1,-1
 122160  177777
 122162  177777  177777                 TTCTB2: .WORD   -1,-1
 122166                                 TTC10:
 122166  104000                                 EMT                         ;
 122170                                 TTCDONE:
 122170  004767  002376                         JSR     PC,.RSET            ;GO INITIALIZE THE FPS AND STACK; AND
                                                                            ;SEE IF THE USER HAS EXPRESSED
                                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                            ;THE USER TYPED CONTROL G?).

                                        ;*******************************************************************************
                                        ;TEST 601          DESTINATION MODES, MODE 2 (FL=1), TEST
                                        ;*******************************************************************************
 122174                                 TS601:
 122174  012700  000300                         MOV     #300,R0             ;SET UP FPS.
 122200  170100                                 LDFPS   R0
 122202  012700  122226                         MOV     #UUCTP1,R0          ;SET UP THE AC0 OPERAND.
 122206  172410                                 LDD     (R0),AC0
 122210  012700  122240                         MOV     #UUCBF0,R0

 122214  175420                         UUC2:   STCDL   AC0,(R0)+           ;TEST INSTRUCTION.

 122216  020027  122244                         CMP     R0,#UUCBF0+4        ;IS R0 CORRECT?
```

```
122222  001411                          BEQ     UUCDONE
122224  104000                          EMT                     ;
                                ;TEST DATA BUFFER:
122226  000000  000000  000000  UUCTP1: .WORD   0,0,0,0
122234  000000
122236  177777                          -1
122240  177777  177777  177777  UUCBF0: .WORD   -1,-1,-1

122246                          UUCDONE:
122246  004767  002320                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*************************************************************************
                                ;TEST 602       DESTINATION MODES, MODE 4 (FL=1), TEST
                                ;*************************************************************************
122252                          TS602:

122252  012700  000300                  MOV     #300,R0         ;SET UP FPS.
122256  170100                          LDFPS   R0
122260  012700  122304                  MOV     #VVCTP1,R0      ;SET UP THE AC0 OPERAND.
122264  172410                          LDD     (R0),AC0
122266  012700  122322                  MOV     #VVCBF0+4,R0

122272  175440                  VVC2:   STCDL   AC0,-(R0)               ;TEST INSTRUCTION.

122274  020027  122316                  CMP     R0,#VVCBF0      ;IS R0 CORRECT?
122300  001411                          BEQ     VVCDONE
122302  104000                          EMT                     ;
                                ;TEST DATA BUFFER:
122304  000000  000000  000000  VVCTP1: .WORD   0,0,0,0
122312  000000
122314  177777                          -1
122316  177777  177777  177777  VVCBF0: .WORD   -1,-1,-1

122324                          VVCDONE:
122324  004767  002242                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).

                                ;*************************************************************************
                                ;TEST 603       STCDI AND STCDL TEST
                                ;*************************************************************************
122330                          TS603:

                                ;FIRST TEST STC WITH EXP=100 (EXCESS 200)
122330  004737  123266          WWC1:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
122334  020000  000000  000000  1$:     .WORD   20000,0,0,0     ;AC0 OPERAND.
122342  000000
122344  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
122350  177777  177777          3$:     .WORD   -1,-1           ;ERROR RES.
122354  040300                  4$:     40300                   ;FPS BEFORE EXECUTION.
```

```
22183 122356 040304                               40304                   ;FPS AFTER EXECUTION.
22184 122360 140304                               140304                  ;ANTICIPATED ERRONEOUS FPS.
22185 122362 177777                               -1               ;REPORT RESULT INCORRECT.
22186                                     ;EXP=0 (OCT)    FL=1    FIC=0
22187 122364 004737 123266              WWC2:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
22188 122370 040000 000000 000000       1$:      .WORD   40000,0,0,0      ;AC     ;AC0 OPERAND.
22189 122376 000000
22190 122400 000000 000000              2$:      .WORD   0,0              ;EXPECTED RESULT.
22191 122404 177777 177777              3$:      .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
22192 122410 040313                      4$:      40313                    ;FPS BEFORE EXECUTION.
22193 122412 040304                               40304                    ;FPS AFTER EXECUTION.
22194 122414 140304                               140304                   ;ANTICIPATED ERRONEOUS FPS.
22195 122416 177777                               -1               ;EXPECTED FEC.
22196                                     ;EXP=37 (OCT)   FL=1    FIC=1
22197 122420 004737 123266              WWC3:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
22198 122424 047667 075757 157737       1$:      .WORD   47667,75757,157737,167773       ;AC0 OPERAND.
22199 122432 167773
22200 122434 055675 173757              2$:      .WORD   55675,173757             ;EXPECTED RESULT.
22201 122440 122102 004021              3$:      .WORD   122102,004021            ;ANTICIPATED ERRONEOUS RESULT.
22202 122444 040717                      4$:      40717                    ;FPS BEFORE EXECUTION.
22203 122446 040700                               40700                    ;FPS AFTER EXECUTION.
22204 122450 140705                               140705                   ;ANTICIPATED ERRONEOUS FPS.
22205 122452 177777                               -1               ;EXPECTED FEC.
22206                                     ;EXP=40 (OCT)   FL=1    FIC=1
22207 122454 004737 123266              WWC4:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
22208 122460 050000 000000 000000       1$:      .WORD   50000,0,0,0      ;AC0 OPERAND.
22209 122466 000000
22210 122470 000000 000000              2$:      .WORD   0,0              ;EXPECTED RESULT.
22211 122474 177777 177777              3$:      .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
22212 122500 040700                      4$:      40700                    ;FPS BEFORE EXECUTION.
22213 122502 140705                               140705                   ;FPS AFTER EXECUTION.
22214 122504 040705                               40705                    ;ANTICIPATED ERRONEOUS FPS.
22215 122506 000006                               6                ;EXPECTED FEC.
22216
22217                                     ;EXP=40 (OCT)   FL=1    FIC=0
22218 122510 004737 123266              WWC5:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
22219 122514 050000 000000 000000       1$:      .WORD   50000,0,0,0      ;AC0 OPERAND.
22220 122522 000000
22221 122524 000000 000000              2$:      .WORD   0,0              ;EXPECTED RESULT.
22222 122530 177777 177777              3$:      .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
22223 122534 040312                      4$:      40312                    ;FPS BEFORE EXECUTION.
22224 122536 040305                               40305                    ;FPS AFTER EXECUTION.
22225 122540 140305                               140305                   ;ANTICIPATED ERRONEOUS FPS.
22226 122542 177777                               -1               ;EXPECTED FEC.
22227                                     ;EXP=30 (OCT)   FL=1    FIC=1
22228 122544 004737 123266              WWC6:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
22229 122550 046000 000001 000000       1$:      .WORD   46000,1,0,0      ;AC0 OPERAND.
22230 122556 000000
22231 122560 000200 000001              2$:      .WORD   200,1            ;EXPECTED RESULT.
22232 122564 177777 177777              3$:      .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
22233 122570 040700                      4$:      40700                    ;FPS BEFORE EXECUTION.
22234 122572 040700                               40700                    ;FPS AFTER EXECUTION.
22235 122574 177777                               -1                       ;ANTICIPATED ERRONEOUS FPS.
22236 122576 177777                               -1               ;EXPECTED FEC.
22237                                     ;EXP=27 (OCT)   FL=1    FIC=1
22238 122600 004737 123266              WWC7:    JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
```

```
122604  045600  000001  000000  1$:     .WORD   45600,1,0,0         ;AC0 OPERAND.
122612  000000
122614  000100  000000          2$:     .WORD   100,0               ;EXPECTED RESULT.
122620  177777  177777          3$:     .WORD   -1,-1               ;ANTICIPATED ERRONEOUS RESULT.
122624  040707                  4$:     40707                       ;FPS BEFORE EXECUTION.
122626  040700                          40700                       ;FPS AFTER EXECUTION.
122630  177777                          -1                          ;ANTICIPATED ERRONEOUS FPS.
122632  177777                          -1                  ;EXPECTED FEC.

                                ;EXP=17 (OCT)   FL=0     FIC=1
122634  004737  123266          WWC10:  JSR     PC,@#STCSUB         ;GO EXECUTE THE INSTRUCTION.
122640  043600  000000  000000  1$:     .WORD   43600,0,0,0         ;AC0 OPERAND.
122646  000000
122650  040000  177777          2$:     .WORD   40000,-1                    ;EXPECTED RESULT.
122654  000000  177777          3$:     .WORD   0,-1                ;ANTICIPATED ERRONEOUS RESULT.
122660  040600                  4$:     40600                       ;FPS BEFORE EXECUTION.
122662  040600                          40600                       ;FPS AFTER EXECUTION.
122664  140604                          140604                      ;ANTICIPATED ERRONEOUS FPS.
122666  177777                          -1                  ;EXPECTED FEC.


                                ;EXP=20 (OCT)   FL=0     FIC=1
122670  004737  123266          WWC11:  JSR     PC,@#STCSUB         ;GO EXECUTE THE INSTRUCTION.
122674  044000  000000  000000  1$:     .WORD   44000,0,0,0         ;AC0 OPERAND.
122702  000000
122704  000000  177777          2$:     .WORD   0,-1                ;EXPECTED RESULT.
122710  177777  177777          3$:     .WORD   -1,-1               ;ANTICIPATED ERRONEOUS RESULT.
122714  040600                  4$:     40600                       ;FPS BEFORE EXECUTION.
122716  140605                          140605                      ;FPS AFTER EXECUTION.
122720  040600                          40600                       ;ANTICIPATED ERRONEOUS FPS.
122722  000006                          6                   ;EXPECTED FEC.
                                ;EXP=10 (OCT), AC NEGATIVE, FL=0, FIC=1
122724  004737  123266          WWC12:  JSR     PC,@#STCSUB         ;GO EXECUTE THE INSTRUCTION.
122730  142000  000000  000000  1$:     .WORD   142000,0,0,0        ;AC0 OPERAND.
122736  000000
122740  177600  177777          2$:     .WORD   177600,-1                   ;EXPECTED RESULT.
122744  000200  000000          3$:     .WORD   200,0               ;ANTICIPATED ERRONEOUS RESULT.
122750  040600                  4$:     40600                       ;FPS BEFORE EXECUTION.
122752  040610                          40610                       ;FPS AFTER EXECUTION.
122754  040600                          40600                       ;ANTICIPATED ERRONEOUS FPS.
122756  177777                          -1                  ;EXPECTED FEC.
                                ;EXP=37 (OCT), FL=1, FIC=1, AC NEG.
122760  004737  123266          WWC13:  JSR     PC,@#STCSUB         ;GO EXECUTE THE INSTRUCTION.
122764  147600  000000  000000  1$:     .WORD   147600,0,0,0        ;AC0 OPERAND.
122772  000000
122774  140000  000000          2$:     .WORD   140000,0                    ;EXPECTED RESULT.
123000  137777  000000          3$:     .WORD   137777,0                    ;ANTICIPATED ERRONEOUS RESULT.
123004  040700                  4$:     40700                       ;FPS BEFORE EXECUTION.
123006  040710                          40710                       ;FPS AFTER EXECUTION.
123010  177777                          -1                          ;ANTICIPATED ERRONEOUS FPS.
123012  177777                          -1                  ;EXPECTED FEC.
                                ;EXP=37  (OCT),  FL=1,  FIC=1,  AC  NEG
123014  004737  123266          WWC14:  JSR     PC,@#STCSUB         ;GO EXECUTE THE INSTRUCTION.
123020  147600  000000  001000  1$:     .WORD   147600,0,1000,0     ;AC0 OPERAND.
123026  000000
123030  137777  177777          2$:     .WORD   137777,177777               ;EXPECTED RESULT.
123034  140000  177777          3$:     .WORD   140000,177777               ;ANTICIPATED ERRONEOUS RESULT.
123040  040707                  4$:     40707                       ;FPS BEFORE EXECUTION.
```

```
123042  040710                          40710                           ;FPS AFTER EXECUTION.
123044  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
123046  177777                          -1                    ;EXPECTED FEC.
                                ;EXP=41 (OCT),  AC NEG, FL=1, FIC=1
123050  004737  123266          WWC15:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123054  150200  000000  000000  1$:     .WORD   150200,0,0,0    ;AC0 OPERAND.
123062  000000
123064  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
123070  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
123074  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
123076  140705                          140705                  ;FPS AFTER EXECUTION.
123100  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
123102  000006                          6                 ;EXPECTED FEC.
                                ;EXP=40 (OCT),  AC NEG,  FL=1,  FIC=1
123104  004737  123266          WWC16:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123110  150000  000001  000000  1$:     .WORD   150000,1,0,0    ;AC0 OPERAND.
123116  000000
123120  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
123124  100000  177600          3$:     .WORD   100000,-200             ;ANTICIPATED ERRONEOUS RESULT.
123130  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
123132  140705                          140705                  ;FPS AFTER EXECUTION.
123134  040700                          40700                   ;ANTICIPATED ERRONEOUS FPS.
123136  000006                          6                 ;EXPECTED FEC.
                                ;EXP=40, AC NEGATIVE, FL=1, FIC=1
123140  004737  123266          WWC17:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123144  150001  000000  000000  1$:     .WORD   150001,0,0,0    ;AC0 OPERAND.
123152  000000
123154  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
123160  077400  000000          3$:     .WORD   77400,0         ;ANTICIPATED ERRONEOUS RESULT.
123164  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
123166  140705                          140705                  ;FPS AFTER EXECUTION.
123170  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
123172  000006                          6                 ;EXPECTED FEC.
                                ;EXP 40    (OCT), AC MOST NEG LONG INT, FL=1
                                ;FIC=1
123174  004737  123266          WWC20:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123200  150000  000000  000000  1$:     .WORD   150000,0,0,0    ;AC0 OPERAND.
123206  000000
123210  100000  000000          2$:     .WORD   100000,0                ;EXPECTED RESULT.
123214  000000  000000          3$:     .WORD   0,0             ;ANTICIPATED ERRONEOUS RESULT.
123220  040700                  4$:     40700                           ;FPS BEFORE EXECUTION.
123222  040710                          40710                   ;FPS AFTER EXECUTION.
123224  140705                          140705                  ;ANTICIPATED ERRONEOUS FPS.
123226  177777                          -1                ;EXPECTED FEC.
                                ;EXP=20, AC = MOST NEG INTEGER, FL=0, FIC=1

123230  004737  123266          WWC21:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123234  144000  000001  000000  1$:     .WORD   144000,1,0,0,   ;AC0 OPERAND.
123242  000000
123244  100000  177777          2$:     .WORD   100000,-1               ;EXPECTED RESULT.
123250  100000  177400          3$:     .WORD   100000,177400           ;ANTICIPATED ERRONEOUS RESULT.
123254  040600                  4$:     40600                   ;FPS BEFORE EXECUTION.
123256  040610                          40610                   ;FPS AFTER EXECUTION.
123260  140605                          140605                  ;ANTICIPATED ERRONEOUS FPS.
123262  177777                          -1                ;EXPECTED FEC.
123264  000457                  6$:     BR      WWCDONE
```

```
                                ;THIS SUBROUTINE, STCSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
                                ;THE STCDI OR STCDL INSTRUCTION AND CHECK THE RESULTS. A CALL
                                ;TO IT IS MADE THUS:
                                ;
                                ;               JSR     PC,@#STCSUB
                                ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                ;               RES:    .WORD   X,X             ;EXPECTED RESULT
                                ;               ERRES:  .WORD   X,X             ;ERROR RESULT
                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                ;               ERFPS:  .WORD   X               ;ERROR FPS.
                                ;               FEC:    .WORD   X               ;EXPECTED FEC
                                ;               ERR1:   ERROR   X               ;DATA ERROR.
                                ;                       BR      CONT
                                ;               ERR2:   ERROR   X               ;FPS ERROR.
                                ;               CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE STCDI OR STCDL INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCSUB RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCSUB
                                ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCSUB WILL RETURN
                                ;TO THE ERROR CALL AT ERR2, OTHERWISE STCSUB ITSELF
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;STCDI OR STCDL IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCSUB
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCSUB WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

123266  012601                  STCSUB: MOV     (SP)+,R1                ;GET A POINTER TO THE ARGUMENTS.
123270  012700  000200                  MOV     #200,R0                 ;SET UP THE AC0 OPERAND.
123274  170100                          LDFPS   R0
123276  010100                          MOV     R1,R0
123300  172410                          LDD     (R0),AC0
123302  012702  123414                  MOV     #STCIBF,R2              ;INITIALIZE THE OUT PUT BUFFER.
123306  012700  000004                  MOV     #4,R0
123312  012722  177777          1$:     MOV     #-1,(R2)+
123316  077003                          SOB     R0,1$
123320  016100  000020                  MOV     20(R1),R0               ;SET THE FPS.
123324  170100                          LDFPS   R0
123326  012700  123414                  MOV     #STCIBF,R0
123332  175410                  2$:     STCDL   AC0,(R0)                ;TEST INSTRUCTION.

123334  170204                          STFPS   R4                      ;GET THE FPS.
123336  170305                          STST    R5                      ;GET THE FEC.
123340  010102                          MOV     R1,R2
123342  062702  000010                  ADD     #10,R2
123346  012700  123414                  MOV     #STCIBF,R0              ;SEE IF THE RESULT IS CORRECT.
123352  012703  000002                  MOV     #2,R3
123356  022022                  3$:     CMP     (R0)+,(R2)+
123360  001014                          BNE     10$
123362  077303                          SOB     R3,3$
```

```
123364  016102  000022                  MOV     22(R1),R2
123370  020204                          CMP     R2,R4           ;SEE IF THE FPS IS CORRECT.
123372  001007                          BNE     10$             ;BRANCH IF INCORRECT.
123374  005702                          TST     R2
123376  100003                          BPL     4$
123400  026105  000026                  CMP     26(R1),R5       ;SEE IF THE FEC IS CORRECT.
123404  001002                          BNE     10$             ;BRANCH IF INCORRECT.

123406  000161  000030          4$:     JMP     30(R1)          ;RETURN.
123412                          10$:
123412  104000                          EMT                     ;

                                ;DATA BUFFER:
123414  177777  177777  177777  STCIBF: .WORD   -1,-1,-1,-1
123422  177777

123424                          WWCDONE:
123424  004767  001142                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*********************************************************************
                                ;TEST 604       STCFL AND STCFI TEST
                                ;*********************************************************************
123430                          TS604:


                                ;EXPONENT=37, FL=1
123430  004737  123266                  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
123434  047777  177777  177777  1$:     .WORD   47777,-1,-1,-1  ;AC0 OPERAND.
123442  177777
123444  077777  177600          2$:     .WORD   77777,177600            ;EXPECTED RESULT.
123450  077777  177777          3$:     .WORD   77777,177777            ;ANTICIPATED ERRONEOUS RESULT.
123454  040100                  4$:     40100                   ;FPS BEFORE EXECUTION.
123456  040100                          40100                   ;FPS AFTER EXECUTION.
123460  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
123462  177777                          -1              ;EXPECTED FEC.
123464                          XXCDONE:
123464  004767  001102                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).


                                ;*********************************************************************
                                ;TEST 605       STEXP TEST
                                ;*********************************************************************
123470                          TS605:

                                ; EXP = 100 (EXCESS 200)
123470  004737  123676          YYC1:   JSR     PC,@#STXSUB
123474  020000  000000  000000  1$:     .WORD   20000,0,0,0                     ;AC
```

```
123502  000000
123504  177700                          2$:     -100                                    ;EXP RES
123506  052525                          3$:     52525                                   ;ERROR EXP.
123510  040000                          4$:     40000                                   ;FPSB
123512  040010                                  40010                                   ;FPSA
123514  040000                                  40000                                   ;ERROR FPS
                                        ; EXP = 200 (EXCESS 200)
123516  004737  123676                  YYC2:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
123522  040000  000000  000000          1$:     .WORD   40000,0,0,0     ;AC0 OPERAND.
123530  000000
123532  000000                          2$:     0                       ;EXPECTED EXPONENT RESULT.
123534  052525                          3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
123536  040000                          4$:     40000                   ;FPS BEFORE EXECUTION.
123540  040004                                  40004                   ;FPS AFTER EXECUTION.
123542  040000                                  40000                   ;ANTICIPATED ERRONEOUS FPS.
                                        ; EXP = 201 (EXCESS 200)
123544  004737  123676                  YYC3:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
123550  040200  000000  000000          1$:     .WORD   40200,0,0,0     ;AC0 OPERAND.
123556  000000
123560  000001                          2$:     1                       ;EXPECTED EXPONENT RESULT.
123562  052525                          3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
123564  040000                          4$:     40000                   ;FPS BEFORE EXECUTION.
123566  040000                                  40000                   ;FPS AFTER EXECUTION.
123570  040004                                  40004                   ;ANTICIPATED ERRONEOUS FPS.
                                        ; EXP = 375  (EXCESS 200)

123572  004737  123676                  YYC4:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
123576  077200  000000  000000          1$:     .WORD   77200,0,0,0     ;AC0 OPERAND.
123604  000000
123606  000175                          2$:     175                     ;EXPECTED EXPONENT RESULT.
123610  052525                          3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
123612  040000                          4$:     40000                   ;FPS BEFORE EXECUTION.
123614  040000                                  40000                   ;FPS AFTER EXECUTION.
123616  040010                                  40010                   ;ANTICIPATED ERRONEOUS FPS.
                                        ; EXP = 1  (EXCESS 200)

123620  004737  123676                  YYC5:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
123624  000200  000000  000000          1$:     .WORD   200,0,0,0       ;AC0 OPERAND.
123632  000000
123634  177601                          2$:     -177                    ;EXPECTED EXPONENT RESULT.
123636  052525                          3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
123640  040000                          4$:     40000                   ;FPS BEFORE EXECUTION.
123642  040010                                  40010                   ;FPS AFTER EXECUTION.
123644  040000                                  40000                   ;ANTICIPATED ERRONEOUS FPS.
                                        ; EXP = 156  (EXCESS 200)

123646  004737  123676                  YYC6:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
123652  033400  000000  000000          1$:     .WORD   33400,0,0,0     ;AC0 OPERAND.
123660  000000
123662  177756                          2$:     -22                     ;EXPECTED EXPONENT RESULT.
123664  052525                          3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
123666  047707                          4$:     47707                   ;FPS BEFORE EXECUTION.
123670  047710                                  47710                   ;FPS AFTER EXECUTION.
123672  177777                                  -1                      ;ANTICIPATED ERRONEOUS FPS.
123674  000452                                  BR      YYCDONE
```

```
                                        ;THIS SUBROUTINE, STXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
                                        ;THE STEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
                                        ;TO IT IS MADE THUS:
                                        ;
                                        ;               JSR     PC,@#STXSUB
                                        ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
                                        ;               RES:    .WORD   X               ;EXPECTED RESULT
                                        ;               ERRES:  .WORD   X               ;ERROR RESULT
                                        ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                                        ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                                        ;               ERFPS:  .WORD   X               ;ERROR FPS.
                                        ;               ERR1:   ERROR   X               ;DATA ERROR.
                                        ;                       BR      CONT
                                        ;               ERR2:   ERROR   X               ;FPS ERROR.
                                        ;               CONT:                           ;RETURN ADDRESS
                                        ;
                                        ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                        ;THE STEXP INSTRUCTION IS EXECUTED.
                                        ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                        ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STXSUB RETURNS CONTROL
                                        ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STXSUB
                                        ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STXSUB WILL RETURN
                                        ;TO THE ERROR CALL AT ERR2, OTHERWISE STXSUB ITSELF
                                        ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                        ;STEXP IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                        ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                        ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STXSUB
                                        ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                        ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STXSUB WILL
                                        ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

123676  012601                          STXSUB: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
123700  010102                                  MOV     R1,R2
123702  012737  123456  124010                  MOV     #123456,@#STXBF
123710  012737  076543  124012                  MOV     #76543,@#STXBF+2
123716  012700  000200                          MOV     #200,R0
123722  170100                                  LDFPS   R0
123724  010100                                  MOV     R1,R0           ;SET UP THE AC0 OPERAND.
123726  172410                                  LDD     (R0),AC0
123730  016100  000016                          MOV     16(R1),R0       ;SET THE FPS.
123734  170100                                  LDFPS   R0
123736  012700  124010                          MOV     #STXBF,R0
123742  175010                          1$:     STEXP   AC0,(R0)        ;TEST INSTRUCTION.
123744  170204                                  STFPS   R4              ;GET FPS.
123746  026137  000010  124010                  CMP     10(R1),@#STXBF  ;WAS RESULT CORRECT?
123754  001401                                  BEQ     5$
123756  104000                                  EMT                     ;
123760  020461  000016                  5$:     CMP     R4,16(R1)       ;SEE IF THE FPS IS CORRECT.
123764  001401                                  BEQ     10$
123766  104000                                  EMT                     ;
                                        ;SEE IF MORE THAN ONE WORD WAS WRITTEN IN THE OUTPUT BUFFER.
123770  022737  076543  124012          10$:    CMP     #76543,@#STXBF+2
123776  001401                                  BEQ     4$
124000  104000                                  EMT                     ;
124002  000161  000022                  4$:     JMP     22(R1)
```

```
124006  177777                           -1
124010  177777  177777  177777  STXBF:   .WORD   -1,-1,-1,-1,-1
124016  177777  177777
124022                          YYCDONE:
124022  004767  000544                   JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                                   ;SEE IF THE USER HAS EXPRESSED
                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                   ;THE USER TYPED CONTROL G?).
                                ;*************************************************************************
                                ;TEST 606        STST TEST
                                ;*************************************************************************
124026                          TS606:
124026  012700  040000                   MOV     #40000,R0         ;SET FPS. FID=1.
124032  170100                           LDFPS   R0
124034  170003                  ZZC2:    .WORD   170003                 ;ILLEGAL FPP
                                                                        ;OP CODE
124036  012700  124116                   MOV     #ZZCBF,R0         ;SET UP THE OUTPUT BUFFER.
124042  012710  177777                   MOV     #-1,(R0)
124046  012760  177777  000002           MOV     #-1,2(R0)
124054  170310                  ZZC3:    STST    (R0)                   ;GET FEC AND
                                                                        ;FEA
124056  170204                           STFPS   R4                ;GET FPS.
124060  012700  124116                   MOV     #ZZCBF,R0
124064  022710  000002                   CMP     #2,(R0) ;SEE IF FEC IS CORRECT.
124070  001010                           BNE     ZZC10             ;BRANCH IF INCORRECT.
124072  022760  124034  000002           CMP     #ZZC2,2(R0)       ;SEE IF FEA, ADDRESS, IS CORRECT.
124100  001004                           BNE     ZZC10             ;BRANCH IF INCORRECT.
124102  022704  140000                   CMP     #140000,R4        ;SEE IF FPS IS CORRECT.
124106  001001                           BNE     ZZC10             ;BRANCH IF INCORRECT.
124110  000407                           BR      ZZCDONE
124112                          ZZC10:
124112  104000                           EMT                       ;
                                ;DATA BUFFER:
124114  177777                           -1
124116  177777  177777  177777  ZZCBF:   .WORD   -1,-1,-1,-1
124124  177777
124126  177777                           -1
124130                          ZZCDONE:
124130  004767  000436                   JSR     PC,.RSET          ;GO INITIALIZE THE FPS AND STACK; AND
                                                                   ;SEE IF THE USER HAS EXPRESSED
                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                   ;THE USER TYPED CONTROL G?).
                                ;*************************************************************************
                                ;TEST 607        SPECIAL CASE TEST
                                ;*************************************************************************
124134                          TS607:
```

```
124134  012746  144724           AAD1:     MOV     #144724, -(SP)  ;PUT FRACTION ON STACK
124140  012746  040600                     MOV     #40600, -(SP)   ;PUT EXPONENT ON STACK
124144  005046                             CLR     -(SP)           ;PUT SUBTRAHEND FRACTION ON STACK
124146  012746  040600                     MOV     #40600, -(SP)   ;PUT SUBTRAHEND EXPONENT ON STACK
124152  172466  000004                     LDF     4(SP), AC0      ;LOAD FP ACCUMULATORS
124156  173026                             SUBF    (SP)+, AC0      ;DO SUBTRACTION
124160  174037  124210                     STF     AC0, @#AADBF    ;GET AND STORE ANSWER
124164  022737  036711  124210             CMP     #36711, @#AADBF ;IS EXPONENT CORRECT
124172  001401                             BEQ     1$
124174  104000                             EMT                     ;BAD EXPONENT FROM SUBTRACTION
124176  022737  152000  124212   1$:       CMP     #152000, @#AADBF+2 ;IS FRACTION CORRECT
124204  001403                             BEQ     AADDONE
124206  104000                             EMT                     ;FRACTION INCORRECT

124210  000000                   AADBF:    .WORD   0
124212  000000                             .WORD   0

124214  012706  001000           AADDONE:  MOV     #STBOT, SP      ;RESTORE STACK POINTER
124220  004767  000346                     JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                   ;SEE IF THE USER HAS EXPRESSED
                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                   ;THE USER TYPED CONTROL G?).




                                 ;*******************************************************************************
                                 ;TEST 610          INTERRUPTABILITY TEST
                                 ;*******************************************************************************
124224                           TS610:
124224  004567  124160           BBD1:     JSR     R5,CHKAPT
124230  000520                             BR      FPEXIT          ;SKIP TEST IF ON APT AND NOT FIRST PASS

124232  005001                             CLR     R1              ;INITIALIZE A COUPLE OF COUNTERS
124234  005000                             CLR     R0
124236  013767  000064  113114             MOV     @#64,   $TMP0   ;SAVE INTERRUPT VECTOR
124244  013767  000066  113110             MOV     @#66,   $TMP1   ;SAVE INTERRUPT PRIORITY
124252  012737  124324  000064             MOV     #3$,    @#64    ;SET UP INTERRUPT PRIORITY FOR THIS TEST
124260  005037  000066                     CLR     @#66            ;AND PRIORITY
124264  005067  053506                     CLR     PS              ;PUT PROCESSOR PRIORITY AT 0
124270  005067  053272                     CLR     TPB             ;SEND A NULL CHARACTER
124274  105767  053264           1$:       TSTB    TTCSR           ;WAIT FOR DONE TO SET
124300  100375                             BPL     1$
124302  005067  053260                     CLR     TPB             ;SEND A SECOND CHARACTER
124306  052767  000100  053250             BIS     #BIT6,  TTCSR   ;SET INTERRUPT ENABLE
124314  005200                   2$:       INC     R0              ;INCREMENT COUNTER TO GET BASE TIME
124316  001376                             BNE     2$              ;CONTINUE LOOPING UNLESS COUNTER GOES TO 0
124320  000005                             RESET                   ;IF NO INTERRUPT YET KILL IT
124322  104000                             EMT                     ;NO INTERRUPT OCCURRED IN ALLOTTED TIME
124324  166700  000110           3$:       SUB     Y,      R0      ;SUBTRACT TIME FOR FP INSTRUCTION
124330  010067  000106                     MOV     R0,     Z       ;SAVE FIRST TIME
124334  012737  124412  000064             MOV     #7$,    @#64    ;SET UP FOR NEXT INTERRUPT
124342  005100                   4$:       COM     R0              ;MAKE PRE LOOP COUNTER NEGATIVE
124344  005067  053214                     CLR     TTCSR           ;MAKE SURE NO INTERRUPT YET
124350  005067  053212                     CLR     TPB             ;SEND A CHARACTER
124354  105767  053204           5$:       TSTB    TTCSR           ;WAIT FOR READY BIT TO SET
```

```
124360  100375                            BPL     5$
124362  005067  053200                    CLR     TPB                  ;SEND SECOND CHARACTER
124366  052767  000100  053170            BIS     #BIT6,   TTCSR       ;SET INTERRUPT ENABLE
124374  005200                    6$:     INC     R0                   ;DO PRE LOOP
124376  001376                            BNE     6$
124400  171227  040400                    MULF    #2,      AC2         ;DO FLOATING POINT INSTRUCTION
124404  000240                            NOP                          ;JUST IN CASE INTERRUPT TAKES TOO LONG
124406  000005                            RESET                        ;IF NO INTERRUPT CLEAR THE WORLD
124410  104000                            EMT                          ;INTERRUPT NOT BACK IN ALLOTTED TIME
124412  005201                    7$:     INC     R1                   ;INCREMENT TIMES THROUGH COUNTER
124414  020127  000015                    CMP     R1,      #15         ;HAVE WE PASSED HERE 15 TIMES BEFORE
124420  001411                            BEQ     BBDDONE              ;IF YES I MAY NEVER PASS HERE AGAIN
124422  062767  000002  000012            ADD     #2,      Z           ;IF NO ADD A LITTLE TIME TO PRELOOP
124430  016700  000006                    MOV     Z,       R0          ;PUT NEW COUNT IN COUNTER
124434  000742                            BR      4$                   ;DO IT ALL AGAIN

124436  000000                    X:      .WORD   0
124440  000026                    Y:      .WORD   26
124442  000000                    Z:      .WORD   0

124444  042767  000100  053112    BBDDONE:BIC     #100,    TTCSR       ;CLEAR INTERRUPT ENABLE BEFORE EXITING TEST
124452  016737  112702  000064            MOV     $TMP0,   @#64        ;RESTORE PRINTER VECTOR
124460  016737  112676  000066            MOV     $TMP1,   @#66        ;RESTORE PRINTER PRIORITY
124466  004767  000100                    JSR     PC,.RSET             ;GO INITIALIZE THE FPS AND STACK; AND
                                                                       ;SEE IF THE USER HAS EXPRESSED
                                                                       ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                       ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                       ;THE USER TYPED CONTROL G?).
124472  000167  000160            FPEXIT: JMP     SLU1ST               ;GET OVER SUBROUTINES TO NEXT TEST


124476  012737  000003  001002    ERROR4: MOV     #3,@#$FATAL                           ;SET UP FATAL ERROR NUMBER
124504  012767  000001  054266            MOV     #1,$MSGTY                             ;SET FATAL ERROR FLAG
124512  032737  000001  001020            BIT     #1,@#$ENV                             ;UNDER APT
124520  001004                            BNE     FPHLT                                 ;YES
124522  012700  124534                    MOV     #FPMSG,R0
124526  004767  006576                    JSR     PC,TYPE
124532  000777                    FPHLT:  BR      .                    ;STAY HERE FOREVER

124534  040506  046111  042105    FPMSG:  .ASCIZ  /FAILED DURING THE FPP TESTS/<12><15>
124542  042040  051125  047111
124550  020107  044124  020105
124556  050106  020120  042524
124564  052123  005123  000015
                                  .EVEN


                                          .SBTTL  FLAG RESET ROUTINE
                                  ;***************************************************************************************
                                  ;*THIS ROUTINE WILL BE CALLED AT THE END OF EACH FLOATING POINT TEST
                                  ;*TO RESET THE STACK, CLEAR THE FPS AND REINITIALIZE TRAP VECTORS

124572  012737  124476  000244    .RSET:  MOV     #ERROR4,@#FPVECT
124600  012737  021336  000004            MOV     #T04,@#ERRVECT
124606  012737  021340  000010            MOV     #T010,@#10
124614  011600                            MOV     (SP),R0
124616  012706  001000                    MOV     #STBOT,SP
```

```
124622  005004                                    CLR     R4
124624  170104                                    LDFPS   R4
124626  000110                                    JMP     (R0)

                                          ;THESE ARE SOME EQUATES USED IN THE  PROGRAM
        000001                                    BIT0=000001
        000002                                    BIT1=000002
        000004                                    BIT2=000004
        000010                                    BIT3=000010
        000020                                    BIT4=000020
        000040                                    BIT5=000040
        000100                                    BIT6=000100
        000200                                    BIT7=000200
        000400                                    BIT8=000400
        001000                                    BIT9=001000
        002000                                    BIT10=002000
        004000                                    BIT11=004000
        010000                                    BIT12=010000
        020000                                    BIT13=020000
        040000                                    BIT14=040000
        100000                                    BIT15=100000

124630  177560                            RCSR:   177560                  ;ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
124632  177562                            RBUF:   177562                  ;ADDRESS OF RECEIVER BUFFER
124634  177564                            TCSR:   177564                  ;ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
124636  177566                            TBUF:   177566                  ;ADDRESS OF TRANSMITTER BUFFER
124640  000060                            RVECT:  60                      ;RECEIVER INTERRUPT VECTOR
124642  000062                            RPSW:   62
124644  000064                            TVECT:  64                      ;TRANSMITTER INTERRUPT VECTOR
124646  000066                            TPSW:   66

                                          ;REAL TIME CLOCK REGISTER AND VECTOR ADDRESSES
124650  177546                            LKS:    .WORD   177546
124652  000100                            RTCVT:  .WORD   100
124654  000102                            RTCPSW: .WORD   102

124656  000244                            SLU1ST: CLZ
124660  032777  000004  074516                    BIT     #4,@SWR
124666  001402                                    BEQ     1$
124670  000167  001106                            JMP     KWSTRT
124674  012737  000004  001004            1$:     MOV     #4,@#$TESTN     ;PUT TEST NUMBER IN MAILBOX
124702  012737  125720  000030                    MOV     #ERROR5,@#30    ;SET UP FOR CORRECT ERROR CALL


                                          ;*******************************************************************************
                                          ;TEST 611         TEST ABILITY TO REFERENCE TCSR
                                          ;*******************************************************************************
124710                                    TS611:
124710  013703  000004                            MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
124714  012737  124730  000004                    MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
124722  005777  177706                            TST     @TCSR           ;REFERENCE THE XMIT COMMAND/STATUS REG.
124726  000401                                    BR      4$
124730                                    1$:
124730  104000                                    EMT                     ;
124732  010337  000004                    4$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
```

```


                                    ;*************************************************************************
                                    ;TEST 612          TEST ABILITY TO REFERENCE TBUF
                                    ;*************************************************************************
124736                              TS612:
124736  013703  000004                      MOV     @#4,R3              ;SAVE TIMEOUT VECTOR
124742  012737  124756  000004              MOV     #1$,@#4             ;SET UP TIMEOUT VECTOR
124750  005777  177662                      TST     @TBUF               ;REFERENCE THE XMIT BUFFER
124754  000401                              BR      4$
124756                              1$:
124756  104000                              EMT                         ;
124760  010337  000004              4$:     MOV     R3,@#4              ;RESTORE TIMEOUT VECTOR


                                    ;*************************************************************************
                                    ;TEST 613          TEST THAT TCSR BIT7(DONE) CLEARS WHEN XBUF IS LOADED
                                    ;*************************************************************************
124764                              TS613:
124764  032737  000001  001020              BIT     #1,@#$ENV           ;ARE WE RUNNING UNDER APT
124772  001405                              BEQ     70$                 ;IF NO THEN SERIES OF TESTS
124774  005737  001006                      TST     @#$PASS             ;IS THIS FIRST PASS
125000  001402                              BEQ     70$                 ;IF YES THEN DO SERIES OF TESTS
125002  000167  000774                      JMP     KWSTRT              ;IF NO THEN BYPASS SERIES OF TESTS
125006  005077  177624              70$:    CLR     @TBUF               ;LOAD XBUF
125012  105777  177616                      TSTB    @TCSR               ;CHECK DONE
125016  100006                              BPL     3$                  ;BR IF CLEAR
                                                                        ;FILL SECOND BUFFER BECUASE REFRESH COULD CAUSE
                                                                        ;FIRST TEST TO FAIL
125020  005077  177612                      CLR     @TBUF               ;FILL DOUBLE BUFFER
125024  105777  177604                      TSTB    @TCSR               ;CHECK DONE
125030  100001                              BPL     3$
125032  104000                              EMT                         ;
125034  005000                      3$:     CLR     R0                  ;CLEAR TIMER
125036  105777  177572              4$:     TSTB    @TCSR               ;CHECK FOR XMIT DONE
125042  100403                              BMI     5$                  ;IF DONE SETS, BR TO END OF TEST
125044  005200                              INC     R0                  ;INCREMENT TIMER
125046  001373                              BNE     4$
125050  104000                              EMT                         ;
125052                              5$:


                                    ;*************************************************************************
                                    ;TEST 614          TEST THAT TCSR "DONE" SETS WITH RESET
                                    ;*************************************************************************
125052                              TS614:
125052  005077  177560                      CLR     @TBUF               ;LOAD TRANSMIT BUFFER
125056  105777  177552              1$:     TSTB    @TCSR               ;WAIT FOR DONE
125062  100375                              BPL     1$
125064  005077  177546                      CLR     @TBUF               ;LOAD SECOND BUFFER
125070  000240                              NOP
125072  000005                              RESET                       ;SET DONE WITH RESET
125074  105777  177534                      TSTB    @TCSR               ;CHECK FOR DONE SET
125100  100401                              BMI     TS615
125102  104000                              EMT                         ;

```

```
                                      ;*************************************************************************
                                      ;TEST 615         TEST ABILITY TO ACCESS RCSR
                                      ;*************************************************************************
125104                                TS615:
125104  013703  000004                        MOV     @#4,R3            ;SAVE TIMEOUT VECTOR
125110  012737  125124  000004                MOV     #1$,@#4           ;SET UP TIMEOUT VECTOR
125116  005777  177506                        TST     @RCSR             ;ACCESS RCSR
125122  000401                                BR      2$
125124                                1$:
125124  104000                                EMT                       ;
125126  010337  000004                2$:     MOV     R3,@#4            ;RESTORE TIMEOUT VECTOR


                                      ;*************************************************************************
                                      ;TEST 616         TEST ABILITY TO ACCESS RBUF
                                      ;*************************************************************************
125132                                TS616:
125132  013703  000004                        MOV     @#4,R3            ;SAVE TIMEOUT VECTOR
125136  012737  125152  000004                MOV     #1$,@#4           ;SET UP TIMEOUT VECTOR
125144  005777  177462                        TST     @RBUF             ;ACCESS RBUF
125150  000401                                BR      2$
125152                                1$:
125152  104000                                EMT                       ;
125154  010337  000004                2$:     MOV     R3,@#4            ;RESTORE TIMEOUT VECTOR




                                      ;*************************************************************************
                                      ;TEST 617         TEST THAT BIT6 OF RCSR CAN BE SET & RESET
                                      ;*************************************************************************
125160                                TS617:
125160  017703  177454                        MOV     @RVECT,R3         ;SAVE RECEIVE VECTOR
125164  012777  125206  177446                MOV     #1$,@RVECT        ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
125172  106427  000340                        MTPS    #340              ;SET PSW TO PRIORITY 7
125176  032777  000100  177424                BIT     #BIT6,@RCSR       ;TEST BIT6 OF RCSR
125204  001401                                BEQ     2$
125206                                1$:
125206  104000                                EMT                       ;
125210  052777  000100  177412        2$:     BIS     #BIT6,@RCSR       ;SET BIT6 OF RCSR
125216  032777  000100  177404                BIT     #BIT6,@RCSR       ;TEST BIT6 OF RCSR
125224  001001                                BNE     3$
125226  104000                                EMT                       ;
125230  042777  000100  177372        3$:     BIC     #BIT6,@RCSR       ;CLEAR BIT6 OF RCSR
125236  032777  000100  177364                BIT     #BIT6,@RCSR       ;TEST BIT6 OF RCSR
125244  001401                                BEQ     4$
125246  104000                                EMT                       ;
125250                                4$:
125250  052777  000100  177352                BIS     #BIT6,@RCSR       ;SET BIT6 OF RCSR
125256  000005                                RESET                     ;CLEAR BIT6OF RCSR WITH RESET
125260  032777  000100  177342                BIT     #BIT6,@RCSR       ;TEST BIT6 OF RCSR
125266  001401                                BEQ     5$
125270  104000                                EMT                       ;
125272  010377  177342                5$:     MOV     R3,@RVECT         ;RESTORE RECEIVE VECTOR
```

```



                                         ;*******************************************************************************
                                         ;TEST 620        TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
                                         ;*******************************************************************************
125276                                   TS620:
125276  042777  000100  177330                   BIC     #BIT6,@TCSR         ;CLEAR TRANSMIT INTERRUPT ENABLE
125304  017703  177334                           MOV     @TVECT,R3           ;SAVE XMIT VECTOR
125310  012777  125332  177326                   MOV     #2$,@TVECT          ;POINT XMIT VECTOR TO ERROR REPORT
125316  105777  177312                   1$:     TSTB    @TCSR               ;WAIT FOR DONE
125322  100375                                   BPL     1$
125324  106427  000140                           MTPS    #140                ;SET PSW TO PRIORITY 3
125330  000401                                   BR      3$
125332                                   2$:
125332  104000                                   EMT                         ;
125334  012777  125354  177302           3$:     MOV     #4$,@TVECT          ;SET XMIT VECTOR TO END OF TEST
125342  052777  000100  177264                   BIS     #BIT6,@TCSR         ;ENABLE INTERRUPTS
125350  000240                                   NOP

125352  104000                                   EMT                         ;XMIT DID NOT INTERRUPT

125354  042777  000100  177252           4$:     BIC     #BIT6,@TCSR         ;DISABLE INTERRUPTS
125362  022626                                   CMP     (SP)+,(SP)+         ;RESTORE SP AFTER INTERRUPT
125364  010377  177254                           MOV     R3,@TVECT           ;RESTORE XMIT VECTOR


                                         ;*******************************************************************************
                                         ;TEST 621        TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED
                                         ;*******************************************************************************
125370                                   TS621:
125370  042777  000100  177236                   BIC     #BIT6,@TCSR         ;DISABLE INTERRUPTS
125376  106427  000340                           MTPS    #340                ;SET PSW TO PRIORITY 7
125402  017703  177236                           MOV     @TVECT,R3           ;SAVE XMIT VECTOR
125406  012777  125434  177230                   MOV     #2$,@TVECT          ;POINT XMIT VECTOR TO ERROR REPORT
125414  105777  177214                   1$:     TSTB    @TCSR               ;WAIT FOR DONE
125420  100375                                   BPL     1$
125422  052777  000100  177204                   BIS     #BIT6,@TCSR         ;ENABLE INTERRUPT
125430  000240                                   NOP
125432  000401                                   BR      3$
125434                                   2$:
125434  104000                                   EMT                         ;
125436  042777  000100  177170           3$:     BIC     #BIT6,@TCSR         ;CLEAR INTERRUPT ENABLE
125444  012777  125462  177172                   MOV     #4$,@TVECT          ;POINT XMIT VECTOR TO ERROR REPORT
125452  106427  000140                           MTPS    #140                ;SET PSW TO PRIORITY 3
125456  000240                                   NOP
125460  000401                                   BR      5$
125462                                   4$:
125462  104000                                   EMT                         ;
125464  010377  177154                   5$:     MOV     R3,@TVECT           ;RESTORE XMIT VECTOR


                                         ;*******************************************************************************
                                         ;TEST 622        TEST TRANSMITTER FOR DOUBLE INTERRUPTS
                                         ;*******************************************************************************
125470                                   TS622:
```

```
125470  042777  000100  177136            BIC     #BIT6,@TCSR       ;CLEAR INTERRUPT ENABLE
125476  017703  177142                    MOV     @TVECT,R3         ;SAVE XMIT VECTOR
125502  017704  177140                    MOV     @TPSW,R4          ;SAVE XMIT PSW VECTOR
125506  012777  125546  177130            MOV     #2$,@TVECT        ;SET UP XMIT VECTOR
125514  012777  000340  177124            MOV     #340,@TPSW        ;SET PIO 7 AFTER INTERRUPT
125522  106427  000140                    MTPS    #140              ;SET PSW TO PRIORITY 3
125526  105777  177102            1$:     TSTB    @TCSR             ;WAIT FOR DONE
125532  100375                            BPL     1$
125534  052777  000100  177072            BIS     #BIT6,@TCSR       ;ENABLE INTERRUPTS
125542  000240                            NOP

125544  104000                            EMT                       ;
                                                                    ;XMIT INTERRUPT DID NOT OCCUR
125546  022626                    2$:     CMP     (SP)+,(SP)+       ;RESTORE SP AFTER INTERRUPT
125550  012777  125574  177066            MOV     #4$,@TVECT        ;POINT XMIT VECTOR TO ERROR
125556  106427  000140                    MTPS    #140              ;SET PSW TO PRIORITY 3
125562  000240                            NOP                       ;GIVE TIME FOR ANY INTERRUPTS
125564  042777  000100  177042            BIC     #BIT6,@TCSR       ;DISABLE INTERRUPTS
125572  000401                            BR      5$
125574                            4$:
125574  104000                            EMT                       ;
125576  010377  177042            5$:     MOV     R3,@TVECT         ;RESTORE XMIT VECTOR
125602  010477  177040                    MOV     R4,@TPSW          ;RESTORE XMIT PSW VECTOR

                                  ;*****************************************************************************
                                  ;TEST 623         TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF
                                  ;*****************************************************************************
125606                            TS623:
125606  042777  000100  177020            BIC     #BIT6,@TCSR       ;DISABLE INTERRUPTS
125614  106427  000340                    MTPS    #340              ;SET PSW TO PRIORITY 7
125620  017703  177020                    MOV     @TVECT,R3         ;SAVE XMIT VECTOR
125624  012777  125674  177012            MOV     #2$,@TVECT        ;POINT XMIT VECTOR TO ERROR
125632  052777  000100  176774            BIS     #BIT6,@TCSR       ;ENABLE INTERRUPTS
125640  005077  176772                    CLR     @TBUF             ;LOAD TBUF
125644  105777  176764            1$:     TSTB    @TCSR             ;WAIT FOR DONE (INTERRUPT)
125650  100375                            BPL     1$
125652  005077  176760                    CLR     @TBUF             ;FILL SECOND BUFFER TO RESET INT.
125656  106427  000140                    MTPS    #140              ;SET PSW TO PRIORITY 3
125662  000240                            NOP                       ;GIVE TIME FOR ANY INTERRUPTS
125664  042777  000100  176742            BIC     #BIT6,@TCSR       ;DISABLE INTERRUPTS
125672  000401                            BR      3$
125674                            2$:
125674  104000                            EMT                       ;
125676  010377  176742            3$:     MOV     R3,@TVECT         ;RESTORE XMIT VECTOR
125702  005000                            CLR     R0                ;INITIALIZE LOOP COUNTER
125704  005200                    4$:     INC     R0                ;INCREMENT LOOP COUNTER
125706  001376                            BNE     4$                ; UNTIL COUNTER = 0
125710  005777  176716                    TST     @RBUF             ;CLEAR RECEIVER BUFFER
125714  000167  000062                    JMP     KWSTRT            ;GET TO NEXT TEST

125720  012737  000004  001002    ERROR5: MOV     #4,@#$FATAL       ;SET UP FATAL ERROR NUMBER
125726  012767  000001  053044            MOV     #1,$MSGTY         ;SET FATAL ERROR FLAG
125734  032737  000001  001020            BIT     #1,@#$ENV                           ;UNDER APR ?
125742  001004                            BNE     SL1HLT
125744  012700  125756                    MOV     #SL1MSG,R0
125750  004767  005354                    JSR     PC,TYPE
```

```
125754  000777                          SL1HLT: BR      .

125756  040506  046111  042105          SL1MSG: .ASCIZ  /FAILED SLU1 TEST/<12><15>
125764  051440  052514  020061
125772  042524  052123  006412
126000     000
        126002                          .EVEN


126002  000244                          KWSTRT: CLZ
126004  032777  000010  073372                  BIT     #10,@SWR
126012  001402                                  BEQ     1$
126014  000167  000730                          JMP     SLU2ST
126020  012737  000005  001004          1$:     MOV     #5,@#$TESTN     ;PUT TEST NUMBER IN MAILBOX
126026  012737  126640  000030                  MOV     #ERROR6,@#30    ;SET UP ERROR CALL

126034                                  LKSTST:
                                        ;*************************************************************************
                                        ;TEST 624       TEST ABILITY TO ACCESS LKS
                                        ;*************************************************************************
126034                                  TS624:
126034  013703  000004                          MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
126040  012737  126054  000004                  MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
126046  005777  176576                          TST     @LKS            ;ACCESS LKS
126052  000401                                  BR      2$
126054                                  1$:
126054  104000                                  EMT                     ;
126056  010337  000004                  2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR

                                        ;*************************************************************************
                                        ;TEST 625       TEST THAT BIT6 OF LKS CAN BE SET & RESET
                                        ;*************************************************************************
126062                                  TS625:
126062  017703  176564                          MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
126066  012777  126110  176556                  MOV     #1$,@RTCVT      ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
126074  106427  000340                          MTPS    #340            ;SET PSW TO PRIORITY 7
126100  032777  000100  176542                  BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
126106  001401                                  BEQ     2$
126110                                  1$:
126110  104000                                  EMT                     ;
126112  052777  000100  176530          2$:     BIS     #BIT6,@LKS      ;SET BIT6 OF LKS
126120  032777  000100  176522                  BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
126126  001001                                  BNE     3$
126130  104000                                  EMT                     ;
126132  042777  000100  176510          3$:     BIC     #BIT6,@LKS      ;CLEAR BIT6 OF LKS
126140  032777  000100  176502                  BIT     #BIT6,@LKS      ;TEST BIT6 OF LK
126146  001401                                  BEQ     4$
126150  104000                                  EMT                     ;
126152  032737  000001  001020          4$:     BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
126160  001403                                  BEQ     70$             ;IF NO THEN DO TEST
126162  005737  001006                          TST     @#$PASS         ;IS THIS FIRST PASS
126166  001011                                  BNE     5$              ;IF NO SKIP TO TEST END
126170                                  70$:
126170  052777  000100  176452                  BIS     #BIT6,@LKS      ;SET BIT6 OF LKS
126176  000005                                  RESET                   ;CLEAR BIT6 OF LKS WITH RESET
126200  032777  000100  176442                  BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
```

```
126206  001401                                  BEQ     5$
126210  104000                                  EMT                             ;
126212  010377  176434                  5$:     MOV     R3,@RTCVT               ;RESTORE LINE CLOCK VECTOR
                                        ;*************************************************************************
                                        ;TEST 626       TEST THAT THE REAL TIME CLOCK INTERRUPTS PROPERLY
                                        ;*************************************************************************
126216                                  TS626:
126216  106427  000340                          MTPS    #340                    ;SET PSW TO PRIORITY 7
126222  017703  176424                          MOV     @RTCVT,R3               ;SAVE LINE CLOCK VECTOR
126226  017704  176422                          MOV     @RTCPSW,R4              ;SAVE LINE CLOCK PSW VECTOR
126232  012777  126274  176412                  MOV     #ESR51,@RTCVT           ;SET RTC INTERRUPT VECTOR TO ERROR REPORT
126240  012777  000340  176406                  MOV     #340,@RTCPSW            ;KEEP PRIORITY AT 7
126246  052777  000100  176374                  BIS     #BIT6,@LKS              ;SET INTERRUPT ENABLE
126254  012701  024000                          MOV     #24000,R1                       ;SET UP A WAIT LOOP
126260  077101                          ESR36:  SOB     R1,ESR36                        ;WAIT 30 MILLISEC
126262  012777  126276  176362                  MOV     #ESR37,@RTCVT                   ;ALTER VECTOR
126270  106427  000240                          MTPS    #240                    ;PRIORITY NOW TO FIVE
126274  104000                          ESR51:  EMT
126276  012777  126314  176346          ESR37:  MOV     #ESR38,@RTCVT
126304  012701  024000                          MOV     #24000,R1
126310  077101                          ESR39:  SOB     R1,ESR39                ;WAIT 30 MORE MILLISEC
126312  000401                                  BR      ESR38+2
126314  104000                          ESR38:  EMT                             ;AN ERROR IF WE ARE HERE
126316  005077  176326                          CLR     @LKS                    ;CLR INTERRUPT ENABLE
126322  012701  024000                          MOV     #24000,R1
126326  077101                          ESR52:  SOB     R1,ESR52                ;WAIT LOOP
126330  106427  000240                          MTPS    #240                            ;ALTER PRIORITY TO FIVE
126334  012701  024000                          MOV     #24000,R1
126340  077101                          ESR53:  SOB     R1,ESR53                ;WAIT AGAIN
126342  012777  126366  176302                  MOV     #ESR55,@RTCVT
126350  052777  000100  176272                  BIS     #BIT6,@LKS                      ;ENABLE LTC INTERRUPTS
126356  012701  024000                          MOV     #24000,R1
126362  077101                          ESR54:  SOB     R1,ESR54
126364  104000                                  EMT                                     ;SHOULD HAVE INTERRUPTED
126366  022626                          ESR55:  CMP     (SP)+,(SP)+             ;RESTORE SP AFTER INTERRUPT
126370  042777  000100  176252                  BIC     #BIT6,@LKS              ;DISABLE INTERRUPTS
126376  010377  176250                          MOV     R3,@RTCVT               ;RESTORE LINE CLOCK VECTOR
126402  010477  176246                          MOV     R4,@RTCPSW              ;RESTORE LINE CLOCK PSW VECTOR
                                        ;*************************************************************************
                                        ;TEST 627       TEST RTC FOR DOUBLE INTERRUPTS
                                        ;*************************************************************************
126406                                  TS627:
126406  032737  000001  001020                  BIT     #1,@#$ENV               ;ARE WE RUNNING UNDER APT
126414  001403                                  BEQ     70$                     ;IF NO THEN DO TEST
126416  005737  001006                          TST     @#$PASS                 ;IS THIS FIRST PASS
126422  001045                                  BNE     TS630                   ;IF NO THEN SHIP TO NEXT TEST
```

```
126424                          70$:
126424  017703  176222                  MOV     @RTCVT,R3               ;SAVE LINE CLOCK VECTOR
126430  017704  176220                  MOV     @RTCPSW,R4              ;SAVE LINE CLOCK PSW VECTOR
126434  012777  126476  176210          MOV     #2$,@RTCVT              ;SET UP RTC INTERRUPT VECTOR
126442  012777  000340  176204          MOV     #340,@RTCPSW            ;DISALLOW INTERRUPTS AFTER THE INTERRUPT
126450  106427  000240                  MTPS    #240                    ;SET PSW TO PRIORITY 5

126454  005000                          CLR     R0
126456  052777  000100  176164          BIS     #BIT6,@LKS              ;ENABLE CLOCK INTERRUPTS
126464  005200                  1$:     INC     R0
126466  005700                          TST     R0
126470  100375                          BPL     1$
126472  000240                          NOP                             ;GIVE TIME FOR ANY INTERRUPT

126474  104000                          EMT                             ;RTC INTERRUPT DID NOT OCCUR

126476  022626                  2$:     CMP     (SP)+,(SP)+             ;RESTORE SP AFTER INTERRUPT
126500  012777  126516  176144          MOV     #3$,@RTCVT              ;POINT RTC VECTOR TO ERROR REPORT
126506  106427  000240                  MTPS    #240                    ;SET PSW TO PRIORITY 5
126512  000240                          NOP                             ;GIVE SOME TIME FOR AN INTERRUPT
126514  000401                          BR      4$
126516                          3$:
126516  104000                          EMT                             ;
126520  042777  000100  176122  4$:     BIC     #BIT6,@LKS              ;DISABLE CLOCK INTERRUPTS
126526  010377  176120                  MOV     R3,@RTCVT               ;RESTORE LINE CLOCK VECTOR
126532  010477  176116                  MOV     R4,@RTCPSW              ;RESTORE LINE CLOCK PSW VECTOR


                                ;*****************************************************************************
                                ;TEST 630       TEST THAT RTC INTERRUPT CLEARS WITH RESET
                                ;*****************************************************************************
126536                          TS630:
126536  032737  000001  001020          BIT     #1, @#$ENV              ;ARE WE RUNNING UNDER APT
126544  001403                          BEQ     70$                     ;IF NO THEN DO TEST
126546  005737  001006                  TST     @#$PASS                 ;IS THIS FIRST PASS
126552  001113                          BNE     TS631                   ;IF NO THEN SHIP TO NEXT TEST
126554                          70$:
126554  106427  000340                  MTPS    #340                    ;SET PSW TO PRIORITY 7
126560  017703  176066                  MOV     @RTCVT,R3               ;SAVE LINE CLOCK VECTOR
126564  012777  126630  176060          MOV     #2$,@RTCVT              ;POINT RTC VECTOR TO ERROR REPORT
126572  005000                          CLR     R0

126574  052777  000100  176046          BIS     #BIT6,@LKS              ;ENABLE CLOCK INTERRUPTS
126602  005200                  1$:     INC     R0
126604  005700                          TST     R0
126606  100375                          BPL     1$
126610  000005                          RESET                           ;CLEAR PENDING INTERRUPT WITH RESET
126612  106427  000240                  MTPS    #240                    ;SET PSW TO PRIORITY 5
126616  000240                          NOP                             ;GIVE TIME FOR ANY INTERRUPT
126620  042777  000100  176022          BIC     #BIT6,@LKS              ;DISALLOW INTERRUPTS
126626  000401                          BR      3$
126630                          2$:
126630  104000                          EMT                             ;
126632  010377  176014          3$:     MOV     R3,@RTCVT               ;RESTORE LINE CLOCK VECTOR
126636  000444                          BR      SLU2ST
```

```


126640 012737 000005 001002  ERROR6: MOV     #5,@#$FATAL                        ;SET UP FATAL ERROR NUMBER
126646 012767 000001 052124          MOV     #1,$MSGTY                          ;SET FATAL ERROR FLAG
126654 032737 000001 001020          BIT     #1,@#$ENV                          ;UNDER APT /
126662 001004                        BNE     LTCHLT
126664 012700 126676                 MOV     #LTCMSG,R0
126670 004767 004434                 JSR     PC,TYPE
126674 000777                LTCHLT: BR      .
126676 040506 046111 051125  LTCMSG: .ASCIZ  /FAILURE DURING LTC TEST/<12><15>
126704 020105 052504 044522
126712 043516 046040 041524
126720 052040 051505 005124
126726 000015
                             .EVEN

                             ;SERIAL LINE UNIT REGISTER AND VECTOR ADDRESSES FOR SLU2

126730 176500                RCSR2:  176500                     ;ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
126732 176502                RBUF2:  176502                     ;ADDRESS OF RECEIVER BUFFER
126734 176504                TCSR2:  176504                     ;ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
126736 176506                TBUF2:  176506                     ;ADDRESS OF TRANSMITTER BUFFER
126740 000300                RVECT2: 300                        ;RECEIVER INTERRUPT VECTOR
126742 000302                RPSW2:  302
126744 000304                TVECT2: 304                        ;TRANSMITTER INTERRUPT VECTOR
126746 000306                TPSW2:  306

126750 000244                SLU2ST: CLZ
126752 032777 000020 072424          BIT     #20,@SWR
126760 001402                        BEQ     1$
126762 000167 002622                 JMP     UNIQUE
126766 012737 000006 001004  1$:     MOV     #6,@#$TESTN        ;PUT TEST NUMBER IN MAILBOX
126774 012737 131504 000030          MOV     #ERROR7,@#30       ;SET UP FOR CORRECT ERROR CALL



                             ;*****************************************************************************
                             ;TEST 631        TEST ABILITY TO REFERENCE TCSR2
                             ;*****************************************************************************
127002                       TS631:
127002 013703 000004                 MOV     @#4,R3             ;SAVE TIMEOUT VECTOR
127006 012737 127022 000004          MOV     #1$,@#4            ;SET UP TIMEOUT VECTOR
127014 005777 177714                 TST     @TCSR2             ;REFERENCE THE XMIT COMMAND/STATUS REG.
127020 000401                        BR      4$
127022                       1$:
127022 104000                        EMT                        ;
127024 010337 000004         4$:     MOV     R3,@#4             ;RESTORE TIMEOUT VECTOR



                             ;*****************************************************************************
                             ;TEST 632        TEST ABILITY TO REFERENCE TBUF2
                             ;*****************************************************************************
127030                       TS632:
127030 013703 000004                 MOV     @#4,R3             ;SAVE TIMEOUT VECTOR
127034 012737 127050 000004          MOV     #1$,@#4            ;SET UP TIMEOUT VECTOR
127042 005777 177670                 TST     @TBUF2             ;REFERENCE THE XMIT BUFFER
```

```
127046  000401                               BR      4$
127050                               1$:
127050  104000                               EMT                     ;
127052  010337  000004               4$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR


                                     ;*******************************************************************************
                                     ;TEST 633        TEST THAT TCSR2 BIT7(DONE) CLEARS WHEN XBUF IS LOADED
                                     ;*******************************************************************************
127056                               TS633:
127056  032737  000001  001020               BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
127064  001403                               BEQ     70$             ;IF NO THEN DO TEST
127066  005737  001006                       TST     @#$PASS         ;IS THIS FIRST PASS
127072  001022                               BNE     TS634           ;IF NO THEN SHIP TO NEXT TEST
127074                               70$:
127074  005077  177636                       CLR     @TBUF2          ;LOAD XBUF
127100  105777  177630                       TSTB    @TCSR2          ;CHECK DONE
127104  100006                               BPL     3$              ;BR IF CLEAR
                                                                     ;FILL SECOND BUFFER BECUASE REFRESH COULD CAUSE
                                                                     ;FIRST TEST TO FAIL
127106  005077  177624                       CLR     @TBUF2          ;FILL DOUBLE BUFFER
127112  105777  177616                       TSTB    @TCSR2          ;CHECK DONE
127116  100001                               BPL     3$
127120  104000                               EMT                     ;
127122  005000                       3$:     CLR     R0              ;CLEAR TIMER
127124  105777  177604               4$:     TSTB    @TCSR2          ;CHECK FOR XMIT DONE
127130  100403                               BMI     5$              ;IF DONE SETS, BR TO END OF TEST
127132  005200                               INC     R0              ;INCREMENT TIMER
127134  001373                               BNE     4$
127136  104000                               EMT                     ;
127140                               5$:


                                     ;*******************************************************************************
                                     ;TEST 634        TEST THAT TCSR2 "DONE" SETS WITH RESET
                                     ;*******************************************************************************
127140                               TS634:
127140  032737  000001  001020               BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
127146  001403                               BEQ     70$             ;IF NO THEN DO TEST
127150  005737  001006                       TST     @#$PASS         ;IS THIS FIRST PASS
127154  001015                               BNE     TS635           ;IF NO THEN SHIP TO NEXT TEST
127156                               70$:
127156  005077  177554                       CLR     @TBUF2          ;LOAD TRANSMIT BUFFER
127162  105777  177546               1$:     TSTB    @TCSR2          ;WAIT FOR DONE
127166  100375                               BPL     1$
127170  005077  177542                       CLR     @TBUF2          ;LOAD SECOND BUFFER
127174  000240                               NOP
127176  000005                               RESET                   ;SET DONE WITH RESET
127200  105777  177530                       TSTB    @TCSR2          ;CHECK FOR DONE SET
127204  100401                               BMI     TS635
127206  104000                               EMT                     ;



                                     ;*******************************************************************************
                                     ;TEST 635        TEST ABILITY TO ACCESS RCSR2
```

```
                                          ;*******************************************************************************
127210                                    TS635:
127210  013703  000004                            MOV     @#4,R3                  ;SAVE TIMEOUT VECTOR
127214  012737  127230  000004                    MOV     #1$,@#4                 ;SET UP TIMEOUT VECTOR
127222  005777  177502                            TST     @RCSR2                  ;ACCESS RCSR
127226  000401                                    BR      2$
127230                                    1$:
127230  104000                                    EMT                             ;
127232  010337  000004                    2$:     MOV     R3,@#4                  ;RESTORE TIMEOUT VECTOR



                                          ;*******************************************************************************
                                          ;TEST 636         TEST ABILITY TO ACCESS RBUF2
                                          ;*******************************************************************************
127236                                    TS636:
127236  013703  000004                            MOV     @#4,R3                  ;SAVE TIMEOUT VECTOR
127242  012737  127256  000004                    MOV     #1$,@#4                 ;SET UP TIMEOUT VECTOR
127250  005777  177456                            TST     @RBUF2                  ;ACCESS RBUF
127254  000401                                    BR      2$
127256                                    1$:
127256  104000                                    EMT                             ;
127260  010337  000004                    2$:     MOV     R3,@#4                  ;RESTORE TIMEOUT VECTOR






                                          ;*******************************************************************************
                                          ;TEST 637         TEST THAT BIT0(BREAK BIT) CAN BE SET & CLEARED & RESET
                                          ;*******************************************************************************
127264                                    TS637:
127264  032777  000001  177442                    BIT     #BIT0,@TCSR2            ;CHECK BIT0 OF TCSR CLEAR
127272  001401                                    BEQ     3$
127274  104000                                    EMT                             ;
127276  052777  000001  177430            3$:     BIS     #BIT0,@TCSR2            ;SET BIT0 IN TCSR
127304  032777  000001  177422                    BIT     #BIT0,@TCSR2            ;TEST BIT0 OF TCSR
127312  001001                                    BNE     4$
127314  104000                                    EMT                             ;
127316  042777  000001  177410            4$:     BIC     #BIT0,@TCSR2            ;CLEAR BIT0 OF TCSR
127324  032777  000001  177402                    BIT     #BIT0,@TCSR2            ;TEST BIT0 OF TCSR
127332  001401                                    BEQ     7$
127334  104000                                    EMT                             ;
127336                                    7$:
127336  032737  000001  001020                    BIT     #1, @#$ENV              ;ARE WE RUNNING UNDER APT .
127344  001403                                    BEQ     70$                     ;IF NO THEN DO TEST
127346  005737  001006                            TST     @#$PASS                 ;IS THIS FIRST PASS
127352  001011                                    BNE     TS640                   ;IF NO THEN SHIP TO NEXT TEST
127354                                    70$:
127354  052777  000001  177352                    BIS     #BIT0,@TCSR2            ;SET BIT0 IN TCSR
127362  000005                                    RESET                           ;CLEAR BIT0 WITH RESET
127364  032777  000001  177342                    BIT     #BIT0,@TCSR2            ;TEST BIT0 CLEAR
127372  001401                                    BEQ     TS640
127374  104000                                    EMT                             ;

```

```
                                            ;*****************************************************************************
                                            ;TEST 640        TEST THAT BIT6(XMIT INT EN) CAN BE SET & RESET
                                            ;*****************************************************************************
127376                                      TS640:
127376  017703  177342                              MOV     @TVECT2,R3              ;SAVE XMIT VECTOR
127402  012777  127424  177334                      MOV     #1$,@TVECT2             ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
127410  106427  000340                              MTPS    #340                    ;SET PSW TO PRIORITY 7
127414  032777  000100  177312                      BIT     #BIT6,@TCSR2            ;TEST BIT6 OF TCSR
127422  001401                                      BEQ     2$
127424                                      1$:
127424  104000                                      EMT                             ;
127426  052777  000100  177300              2$:     BIS     #BIT6,@TCSR2            ;SET BIT6 OF TCSR
127434  032777  000100  177272                      BIT     #BIT6,@TCSR2            ;TEST BIT6 OF TCSR
127442  001001                                      BNE     3$
127444  104000                                      EMT                             ;
127446  042777  000100  177260              3$:     BIC     #BIT6,@TCSR2            ;CLEAR BIT6 OF TCSR
127454  032777  000100  177252                      BIT     #BIT6,@TCSR2            ;TEST BIT6 OF TCSR
127462  001401                                      BEQ     4$
127464  104000                                      EMT                             ;
127466  032737  000001  001020              4$:     BIT     #1,@#$ENV               ;ARE WE RUNNING UNDER APT
127474  001403                                      BEQ     70$                     ;IF NO THEN DO TEST
127476  005737  001006                              TST     @#$PASS                 ;IF THIS FIRST PASS
127502  001011                                      BNE     5$                      ;IF NO THEN SKIP TO END OF TEST
127504                                      70$:
127504  052777  000100  177222                      BIS     #BIT6,@TCSR2            ;SET BIT6 OF TCSR
127512  000005                                      RESET                           ;CLEAR BIT6 WITH RESET
127514  032777  000100  177212                      BIT     #BIT6,@TCSR2            ;TEST BIT6 OF TCSR
127522  001401                                      BEQ     5$
127524  104000                                      EMT                             ;
127526  010377  177212                      5$:     MOV     R3,@TVECT2              ;RESTORE XMIT VECTOR


                                            ;*****************************************************************************
                                            ;TEST 641        TEST THAT BIT6 OF RCSR2 CAN BE SET & RESET
                                            ;*****************************************************************************
127532                                      TS641:
127532  017703  177202                              MOV     @RVECT2,R3              ;SAVE RECEIVE VECTOR
127536  012777  127560  177174                      MOV     #1$,@RVECT2             ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
127544  106427  000340                              MTPS    #340                    ;SET PSW TO PRIORITY 7
127550  032777  000100  177152                      BIT     #BIT6,@RCSR2            ;TEST BIT6 OF RCSR
127556  001401                                      BEQ     2$
127560                                      1$:
127560  104000                                      EMT                             ;
127562  052777  000100  177140              2$:     BIS     #BIT6,@RCSR2            ;SET BIT6 OF RCSR
127570  032777  000100  177132                      BIT     #BIT6,@RCSR2            ;TEST BIT6 OF RCSR
127576  001001                                      BNE     3$
127600  104000                                      EMT                             ;
127602  042777  000100  177120              3$:     BIC     #BIT6,@RCSR2            ;CLEAR BIT6 OF RCSR
127610  032777  000100  177112                      BIT     #BIT6,@RCSR2            ;TEST BIT6 OF RCSR
127616  001401                                      BEQ     4$
127620  104000                                      EMT                             ;
127622  032737  000001  001020              4$:     BIT     #1,@#$ENV               ;ARE WE RUNNING UNDER APT
127630  001403                                      BEQ     70$                     ;IF NO THEN DO TEST
127632  005737  001006                              TST     @#$PASS                 ;IS THIS FIRST PASS
127636  001011                                      BNE     5$                      ;IF NO THEN SKIP TO END OF TEST
127640                                      70$:
```

```
127640  052777  000100  177062          BIS     #BIT6,@RCSR2    ;SET BIT6 OF RCSR
127646  000005                          RESET                   ;CLEAR BIT6OF RCSR2 WITH RESET
127650  032777  000100  177052          BIT     #BIT6,@RCSR2    ;TEST BIT6 OF RCSR
127656  001401                          BEQ     5$
127660  104000                          EMT                     ;
127662  010377  177052          5$:     MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR



                                ;*******************************************************************************
                                ;TEST 642       TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
                                ;*******************************************************************************
127666                          TS642:
127666  042777  000100  177040          BIC     #BIT6,@TCSR2    ;CLEAR TRANSMIT INTERRUPT ENABLE
127674  017703  177044                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
127700  012777  127722  177036          MOV     #2$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
127706  105777  177022          1$:     TSTB    @TCSR2          ;WAIT FOR DONE
127712  100375                          BPL     1$
127714  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
127720  000401                          BR      3$
127722                          2$:
127722  104000                          EMT                     ;
127724  012777  127744  177012  3$:     MOV     #4$,@TVECT2     ;SET XMIT VECTOR TO END OF TEST
127732  052777  000100  176774          BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPTS
127740  000240                          NOP

127742  104000                          EMT                     ;XMIT DID NOT INTERRUPT

127744  042777  000100  176762  4$:     BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
127752  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
127754  010377  176764                  MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR


                                ;*******************************************************************************
                                ;TEST 643       TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED
                                ;*******************************************************************************
127760                          TS643:
127760  042777  000100  176746          BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
127766  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
127772  017703  176746                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
127776  012777  130024  176740          MOV     #2$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
130004  105777  176724          1$:     TSTB    @TCSR2          ;WAIT FOR DONE
130010  100375                          BPL     1$
130012  052777  000100  176714          BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPT
130020  000240                          NOP
130022  000401                          BR      3$
130024                          2$:
130024  104000                          EMT                     ;
130026  042777  000100  176700  3$:     BIC     #BIT6,@TCSR2    ;CLEAR INTERRUPT ENABLE
130034  012777  130052  176702          MOV     #4$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
130042  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
130046  000240                          NOP
130050  000401                          BR      5$
130052                          4$:
130052  104000                          EMT                     ;
130054  010377  176664          5$:     MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
```

```


                                        ;*******************************************************************************
                                        ;TEST 644       TEST TRANSMITTER FOR DOUBLE INTERRUPTS
                                        ;*******************************************************************************
130060                                  TS644:
130060  042777  000100  176646                  BIC     #BIT6,@TCSR2            ;CLEAR INTERRUPT ENABLE
130066  017703  176652                          MOV     @TVECT2,R3              ;SAVE XMIT VECTOR
130072  017704  176650                          MOV     @TPSW2,R4               ;SAVE XMIT PSW VECTOR
130076  012777  130136  176640                  MOV     #2$,@TVECT2             ;SET UP XMIT VECTOR
130104  012777  000340  176634                  MOV     #340,@TPSW2             ;SET PIO 7 AFTER INTERRUPT
130112  106427  000140                          MTPS    #140                    ;SET PSW TO PRIORITY 3
130116  105777  176612                  1$:     TSTB    @TCSR2                  ;WAIT FOR DONE
130122  100375                                  BPL     1$
130124  052777  000100  176602                  BIS     #BIT6,@TCSR2            ;ENABLE INTERRUPTS
130132  000240                                  NOP

130134  104000                                  EMT                             ;
                                                                                ;XMIT INTERRUPT DID NOT OCCUR
130136  022626                          2$:     CMP     (SP)+,(SP)+             ;RESTORE SP AFTER INTERRUPT
130140  012777  130164  176576                  MOV     #4$,@TVECT2             ;POINT XMIT VECTOR TO ERROR
130146  106427  000140                          MTPS    #140                    ;SET PSW TO PRIORITY 3
130152  000240                                  NOP                             ;GIVE TIME FOR ANY INTERRUPTS
130154  042777  000100  176552                  BIC     #BIT6,@TCSR2            ;DISABLE INTERRUPTS
130162  000401                                  BR      5$
130164                                  4$:
130164  104000                                  EMT                             ;
130166  010377  176552                  5$:     MOV     R3,@TVECT2              ;RESTORE XMIT VECTOR
130172  010477  176550                          MOV     R4,@TPSW2               ;RESTORE XMIT PSW VECTOR

                                        ;*******************************************************************************
                                        ;TEST 645       TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF2
                                        ;*******************************************************************************
130176                                  TS645:
130176  032737  000001  001020                  BIT     #1, @#$ENV              ;ARE WE RUNNING UNDER APT
130204  001403                                  BEQ     70$                     ;IF NO THEN DO TEST
130206  005737  001006                          TST     @#$PASS                 ;IS THIS FIRST PASS
130212  001043                                  BNE     TS646                   ;IF NO THEN SHIP TO NEXT TEST
130214                                  70$:
130214  042777  000100  176512                  BIC     #BIT6,@TCSR2            ;DISABLE INTERRUPTS
130222  106427  000340                          MTPS    #340                    ;SET PSW TO PRIORITY 7
130226  017703  176512                          MOV     @TVECT2,R3              ;SAVE XMIT VECTOR
130232  012777  130302  176504                  MOV     #2$,@TVECT2             ;POINT XMIT VECTOR TO ERROR
130240  052777  000100  176466                  BIS     #BIT6,@TCSR2            ;ENABLE INTERRUPTS
130246  005077  176464                          CLR     @TBUF2                  ;LOAD TBUF
130252  105777  176456                  1$:     TSTB    @TCSR2                  ;WAIT FOR DONE (INTERRUPT)
130256  100375                                  BPL     1$
130260  005077  176452                          CLR     @TBUF2                  ;FILL SECOND BUFFER TO RESET INT.
130264  106427  000140                          MTPS    #140                    ;SET PSW TO PRIORITY 3
130270  000240                                  NOP                             ;GIVE TIME FOR ANY INTERRUPTS
130272  042777  000100  176434                  BIC     #BIT6,@TCSR2            ;DISABLE INTERRUPTS
130300  000401                                  BR      3$
130302                                  2$:
130302  104000                                  EMT                             ;
130304  010377  176434                  3$:     MOV     R3,@TVECT2              ;RESTORE XMIT VECTOR
130310  005000                                  CLR     R0                      ;INIT LOOP COUNTER
```

```
130312  005200                          4$:     INC     R0              ;INCREMENT COUNTER
130314  001376                                  BNE     4$              ;UNTIL COUNTER = 0
130316  005777  176410                          TST     @RBUF2          ;CLEAR RECEIVER BUFFER


                                        ;*******************************************************************************
                                        ;TEST 646        TEST THAT RCVR DONE (7) SET & CLEAR PROPERLY
                                        ;*******************************************************************************
130322                                  TS646:
130322  005000                                  CLR     R0              ;CLEAR A TIMER
130324  005077  176406                          CLR     @TBUF2          ;LOAD TRANSMIT BUFFER
130330  105777  176374                  WDONE2: TSTB    @RCSR2          ;CHECK FOR RECEIVER DONE
130334  100403                                  BMI     6$              ;BR, IF DONE
130336  005200                                  INC     R0              ;INCREMENT TIMER, IF NOT DONE
130340  001373                                  BNE     WDONE2
130342  104000                                  EMT                     ;RECEIVER DONE NEVER SET

130344  032737  000001  001020          6$:     BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
130352  001403                                  BEQ     70$             ;IF NO THEN DO TEST
130354  005737  001006                          TST     @#$PASS         ;IS THIS FIRST PASS
130360  001005                                  BNE     2$              ;IF NO THEN SKIP TO END OF TEST
130362                                  70$:
130362  000005                                  RESET                   ;CLEAR DONE WITH RESET
130364  105777  176340                          TSTB    @RCSR2          ;CHECK FOR DONE CLEAR
130370  001401                                  BEQ     2$
130372  104000                                  EMT                     ;RESET DID NOT CLEAR RCVR DONE
130374  005000                          2$:     CLR     R0              ;INIT LOOP COUNTER
130376  005200                          3$:     INC     R0              ;INCREMENT COUNTER
130400  001376                                  BNE     3$              ;UNTIL COUNTER = 0
130402  005777  176324                          TST     @RBUF2          ;CLEAR RECEIVER BUFFER


                                        ;*******************************************************************************
                                        ;TEST 647        TEST THAT READING RBUF2 CLEARS RECEIVER DONE
                                        ;*******************************************************************************
130406                                  TS647:
130406  005077  176324                          CLR     @TBUF2          ;LOAD TRANSMITTER
130412  105777  176312                  1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
130416  100375                                  BPL     1$
130420  017700  176306                          MOV     @RBUF2,R0       ;READ RECEIVE BUFFER
130424  105777  176300                          TSTB    @RCSR2          ;CHECK FOR RECEIVE DONE CLEAR
130430  001401                                  BEQ     TS650
130432  104000                                  EMT                     ;
                                                                        ;READING RBUF2 DID NOT CLEAR RCVR DONE


                                        ;*******************************************************************************
                                        ;TEST 650        TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED
                                        ;*******************************************************************************
130434                                  TS650:
130434  042777  000100  176272                  BIC     #BIT6,@TCSR2    ;DISABLE TRANSMIT INTERRUPTS
130442  042777  000100  176260                  BIC     #BIT6,@RCSR2    ;DISABLE RECEIVER INTERRUPTS
130450  017703  176264                          MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
130454  012777  130502  176256                  MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO ERROR REPORT
130462  106427  000140                          MTPS    #140            ;SET PSW TO PRIORITY 3
130466  005077  176244                          CLR     @TBUF2          ;SEND A CHARACTER
```

```
130472  105777  176232          1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
130476  100375                          BPL     1$
130500  000401                          BR      3$
130502                          2$:
130502  104000                          EMT                     ;
130504  012777  130524  176226  3$:     MOV     #4$,@RVECT2     ;POINT RCV VECTOR TO END OF TEST
130512  052777  000100  176210          BIS     #BIT6,@RCSR2    ;ENABLE RCV INTERRUPTS
130520  000240                          NOP                     ;GIVE ANY INTERRUPTS TIME
130522  104000                          EMT                     ;
130524  042777  000100  176176  4$:     BIC     #BIT6,@RCSR2    ;DISABLE INTERRUPTS
130532  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
130534  005777  176172                  TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
130540  010377  176174                  MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR


                                ;*****************************************************************************
                                ;TEST 651       TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED
                                ;*****************************************************************************
130544                          TS651:
130544  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
130550  017703  176164                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
130554  012777  130606  176156          MOV     #2$,@RVECT2     ;POINT RCVR VECTOR TO ERROR REPORT
130562  005077  176150                  CLR     @TBUF2          ;SEND A CHARACTER
130566  105777  176136          1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
130572  100375                          BPL     1$
130574  052777  000100  176126          BIS     #BIT6,@RCSR2    ;ENABLE INTERRUPTS
130602  000240                          NOP                     ;GIVE TIME FOR INTERRUPT
130604  000401                          BR      3$
130606                          2$:
130606  104000                          EMT                     ;RCVR INTERRUPTS AT PRIORITY 7
130610  042777  000100  176112  3$:     BIC     #BIT6,@RCSR2    ;CLEAR INTERRUPT ENABLE
130616  012777  130634  176114          MOV     #4$,@RVECT2     ;POINT RCVR VECTOR TO ERROR REPORT
130624  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
130630  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
130632  000401                          BR      5$
130634                          4$:
130634  104000                          EMT                     ;RCVR INTERRUPT REQUEST PASSED WITH BIT6 CLEAR
130636  005777  176070          5$:     TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
130642  010377  176072                  MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR


                                ;*****************************************************************************
                                ;TEST 652       TEST RECEIVER FOR DOUBLE INTERRUPTS
                                ;*****************************************************************************
130646                          TS652:
130646  017703  176066                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
130652  017704  176064                  MOV     @RPSW2,R4       ;SAVE RECEIVE PSW VECTOR
130656  012777  130722  176054          MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO CONTINUE TEST
130664  012777  000340  176050          MOV     #340,@RPSW2     ;SET PRIORITY TO 7 AFTER INTERRUPT
130672  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
130676  005077  176034                  CLR     @TBUF2          ;SEND A CHARACTER
130702  105777  176022          1$:     TSTB    @RCSR2          ;WAIT FOR RCVR DONE
130706  100375                          BPL     1$
130710  052777  000100  176012          BIS     #BIT6,@RCSR2    ;ENABLE RCV INTERRUPTS
130716  000240                          NOP                     ;GIVE SOME TIME
130720  104000                          EMT                     ;
```

```
130722  022626                  2$:     CMP     (SP)+,(SP)+         ;RESTORE SP AFTER INTERRUPT
130724  012777  130760  176006          MOV     #3$,@RVECT2         ;POINT RCV VECTOR TO ERROR REPORT
130732  106427  000140                  MTPS    #140                ;SET PSW TO PRIORITY 3
130736  000240                          NOP                         ;GIVE SOME TIME
130740  042777  000100  175762          BIC     #BIT6,@RCSR2        ;CLEAR INTERRUPT ENABLE
130746  010377  175766                  MOV     R3,@RVECT2          ;RESTORE RECEIVE VECTOR
130752  010477  175764                  MOV     R4,@RPSW2           ;RESTORE RECEIVE PSW VECTOR
130756  000401                          BR      4$
130760                          3$:
130760  104000                          EMT                         ;
130762  005777  175744          4$:     TST     @RBUF2              ;CLEAR CHARACTER FROM RECEIVER BUFFER
130766  010377  175746                  MOV     R3,@RVECT2          ;RESTORE RECEIVE VECTOR


                                ;*******************************************************************************
                                ;TEST 653       TEST THAT RCVR INTERRUPT CLEARS BY READING RBUF2
                                ;*******************************************************************************
130772                          TS653:
130772  106427  000340                  MTPS    #340                ;SET PSW TO PRIORITY 7
130776  017703  175736                  MOV     @RVECT2,R3          ;SAVE RECEIVE VECTOR
131002  012777  131052  175730          MOV     #2$,@RVECT2         ;POINT RCV VECTOR TO ERROR REPORT
131010  052777  000100  175712          BIS     #BIT6,@RCSR2        ;SET RCVR INTERRUPT ENABLE
131016  005077  175714                  CLR     @TBUF2              ;SEND A CHARACTER
131022  105777  175702          1$:     TSTB    @RCSR2              ;WAIT FOR DONE (INTERRUPT)
131026  100375                          BPL     1$
131030  005777  175676                  TST     @RBUF2              ;READ RBUF TO CLEAR PENDING INTERRPUT
131034  106427  000140                  MTPS    #140                ;SET PSW TO PRIORITY 3
131040  000240                          NOP                         ;ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
131042  042777  000100  175660          BIC     #BIT6,@RCSR2        ;NO INTERRUPT-CLEAR INT. ENABLE
131050  000401                          BR      3$
131052                          2$:
131052  104000                          EMT                         ;
131054  010377  175660          3$:     MOV     R3,@RVECT2          ;RESTORE RECEIVE VECTOR



                                ;*******************************************************************************
                                ;TEST 654       TEST THAT RESET CLEARS RECEIVE INTERRUPT
                                ;*******************************************************************************
131060                          TS654:
131060  032737  000001  001020          BIT     #1, @#$ENV          ;ARE WE RUNNING UNDER APT
131066  001403                          BEQ     70$                 ;IF NO THEN DO TEST
131070  005737  001006                  TST     @#$PASS             ;IS THIS FIRST PASS
131074  001036                          BNE     TS655               ;IF NO THEN SHIP TO NEXT TEST
131076                          70$:
131076  106427  000340                  MTPS    #340                ;SET PSW TO PRIORITY 7
131102  017703  175632                  MOV     @RVECT2,R3          ;SAVE RECEIVE VECTOR
131106  012777  131164  175624          MOV     #2$,@RVECT2         ;POINT RCV VECTOR TO ERROR REPORT
131114  052777  000100  175606          BIS     #BIT6,@RCSR2        ;SET RCV INTERRUPT ENABLE
131122  012777  000377  175606          MOV     #377,@TBUF2         ;SEND AN ALL 1'S CHARACTER
131130  105777  175574          1$:     TSTB    @RCSR2              ;WAIT FOR RCV DONE
131134  100375                          BPL     1$
131136  000005                          RESET                       ;CLEAR RCV INTERRUPT & RBUF2
131140  052777  000100  175562          BIS     #BIT6,@RCSR2        ;SET RECEIVER INTERRUPT
131146  106427  000140                  MTPS    #140                ;SET PSW TO PRIORITY 3
131152  000240                          NOP                         ;ALLOW TIME FOR AN ERRONEOUS INTERRUPT
```

```
131154  042777  000100  175546                BIC     #BIT6,@RCSR2        ;NO INTERRUPT-CLEAR INT. ENABLE
131162  000401                                BR      3$
131164                                2$:
131164  104000                                EMT                         ;
131166  010377  175546                3$:     MOV     R3,@RVECT2          ;RESTORE RECEIVE VECTOR


                                      ;*******************************************************************************
                                      ;TEST 655        TEST THAT THE "OR" ERROR (BIT14) & "ERROR" (BIT15) CAN BE SET
                                      ;*******************************************************************************
131172                                TS655:
131172  012700  000003                        MOV     #3,R0               ;SET CHARACTER COUNT TO SEND 3 CHAR.
131176  005077  175534                1$:     CLR     @TBUF2              ;LOAD TRANSMIT BUFFER
131202  105777  175526                2$:     TSTB    @TCSR2              ;WAIT FOR TRANSMIT DONE
131206  100375                                BPL     2$
131210  005300                                DEC     R0                  ;DECREMENT CHARACTER COUNT
131212  001371                                BNE     1$                  ;BR IF ALL CHARACTERS NOT TRANSMITTED
131214  032777  040000  175510                BIT     #BIT14,@RBUF2       ;TEST FOR "OR" ERROR FLAG
131222  001001                                BNE     3$
131224  104000                                EMT                         ;
131226  032777  100000  175476        3$:     BIT     #BIT15,@RBUF2       ;TEST "ERROR" FLAG
131234  001001                                BNE     4$
131236  104000                                EMT                         ;
131240  005000                        4$:     CLR     R0                  ;CLEAR LOOP COUNTER
131242  005200                        5$:     INC     R0                  ;INCREMENT LOOP COUNTER
131244  001376                                BNE     5$                  ;  UNTIL COUNTER = 0
131246  005777  175460                        TST     @RBUF2              ;CLEAR CHARACTER FROM RECEIVER BUFFER


                                      ;*******************************************************************************
                                      ;TEST 656        TEST THAT BREAK TRANSMITS ALL ZEROES
                                      ;*******************************************************************************
131252                                TS656:
131252  032737  000001  001020                BIT     #1, @#$ENV          ;ARE WE RUNNING UNDER APT
131260  001403                                BEQ     70$                 ;IF NO THEN DO TEST
131262  005737  001006                        TST     @#$PASS             ;IS THIS FIRST PASS
131266  001027                                BNE     TS657               ;IF NO THEN SHIP TO NEXT TEST
131270                                70$:
131270  012777  177777  175440                MOV     #-1,@TBUF2          ;TRANSMIT ALL ONES TO RCVR
131276  105777  175426                1$:     TSTB    @RCSR2              ;WAIT FOR RCVR DONE
131302  100375                                BPL     1$
131304  005777  175422                        TST     @RBUF2              ;CLEAR DONE (LEAVING ALL ONES IN RBUF)
131310  052777  000001  175416                BIS     #BIT0,@TCSR2        ;TRANSMIT BREAK
131316  005000                                CLR     R0                  ;CLEAR A TIMER
131320  105777  175404                2$:     TSTB    @RCSR2              ;WAIT FOR RCVR DONE
131324  100403                                BMI     CONT42              ;BR IF DONE
131326  005200                                INC     R0                  ;IF NOT, INCREMENT TIMER
131330  001373                                BNE     2$
131332  104000                                EMT                         ;BREAK DID NOT TRANSMIT ANYTHING

131334  105777  175372                CONT42: TSTB    @RBUF2              ;CHECK RECEIVE BUFFER FOR ZERO
131340  001401                                BEQ     3$
131342  104000                                EMT                         ;BREAK DID NOT TRANSMIT ALL ZEROES

131344  000005                        3$:     RESET                       ;CLEAR ERRORS
```

```

                                          ;*******************************************************************************
                                          ;TEST 657          TEST THAT "FR" ERROR CAN BE SET DURING BREAK
                                          ;*******************************************************************************
131346                                    TS657:
131346  052777  000001  175360                    BIS     #BIT0,@TCSR2        ;SEND BREAK
131354  005077  175356                            CLR     @TBUF2              ;TRANSMIT A CHARACTER TO TIME BREAK
131360  105777  175344                    1$:     TSTB    @RCSR2              ;WAIT FOR RCVR DONE
131364  100375                                    BPL     1$
131366  042777  000001  175340                    BIC     #BIT0,@TCSR2        ;CLEAR BREAK BITS
131374  032777  020000  175330                    BIT     #BIT13,@RBUF2       ;CHECK FOR FRAMING ERROR FLAG
131402  001001                                    BNE     2$
131404  104000                                    EMT                         ;BREAK DID NOT SET FRAMING ERROR
131406  032777  100000  175316            2$:     BIT     #BIT15,@RBUF2       ;TEST "ERROR" FLAG
131414  001001                                    BNE     3$
131416  104000                                    EMT                         ;"ERROR" FLAG DID NOT SET WITH "OR" FLAG
131420  005777  175306                    3$:     TST     @RBUF2              ;CLEAR RECEIVER BUFFER

                                          ;*******************************************************************************
                                          ;TEST 660          TEST DATA PATHS USING WRAP CABLE
                                          ;*******************************************************************************
131424                                    TS660:
131424  005001                                    CLR     R1                  ;CLEAR REGISTER FOR TEST DATA
131426  105201                            1$:     INCB    R1                  ;INCREMENT THE TEST DATA
131430  010177  175302                            MOV     R1,@TBUF2           ;XMIT A CHARACTER
131434  005000                                    CLR     R0                  ;CLEAR A TIMER
131436  105777  175266                    2$:     TSTB    @RCSR2              ;WAIT FOR RECEIVER DONE
131442  100403                                    BMI     3$                  ;BR IF DONE
131444  005200                                    INC     R0                  ;INCREMENT TIMER IF NOT
131446  001373                                    BNE     2$
131450  104000                                    EMT                         ;
131452  017702  175254                    3$:     MOV     @RBUF2,R2           ;GET RECEIVED CHARACTER
131456  020102                                    CMP     R1,R2               ;COMPARE DATA
131460  001401                                    BEQ     4$
131462  104000                                    EMT                         ;
131464  105701                            4$:     TSTB    R1                  ;TEST XMIT DATA FOR ZERO
131466  001357                                    BNE     1$                  ;BR, IF NOT FINISHED
131470  000167  000114                            JMP     UNIQUE              ;FINISHED TESTING DEVICES SEPARATELY
                                                                              ;GO TEST THEM ALL TOGETHER
131474  000000                            $BDADR: 0
131476  000000                            $BDDAT: 0
131500  000000                            $GDADR: 0
131502  000000                            $GDDAT: 0


131504  012737  000006  001002            ERROR7: MOV     #6,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
131512  012767  000001  047260                    MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
131520  032737  000001  001020                    BIT     #1,@#$ENV                       ;UNDER APT ?
131526  001004                                    BNE     SL2HLT                          ;YES
131530  012700  131542                            MOV     #SL2MSG,R0
131534  004767  001570                            JSR     PC,TYPE
131540  000777                            SL2HLT: BR      .

131542  040506  046111  051125            SL2MSG: .ASCIZ  /FAILURE DURING SLU 2 TEST/<12><15>
131550  020105  052504  044522
131556  043516  051440  052514
```

```
131564  031040  052040  051505
131572  005124  000015
                                        .EVEN


131576  177560                          DADTBL: .WORD     177560
131600  177564                                  .WORD     177564
131602  176500                                  .WORD     176500
131604  176504                                  .WORD     176504
131606  177564                          TBLEND: .WORD     177564


131610  032777  000034  067566          UNIQUE: BIT       #34,@SWR
131616  001402                                  BEQ       1$
131620  000167  001320                          JMP       ENDPAS
131624  012737  000007  001004          1$:     MOV       #7,@#$TESTN        ;UPDATE TEST NUMBER FOR APT
131632  012737  132450  000030                  MOV       #ERROR8,@#30       ;SET UP FOR CORRECT ERROR CALL

                                        ;*******************************************************************************
                                        ;TEST 661          UNIQUE INTERNAL ADDRESS TEST
                                        ;*******************************************************************************
131640                                  TS661:
131640  032737  000001  001020                  BIT       #1, @#$ENV         ;ARE WE RUNNING UNDER APT
131646  001403                                  BEQ       70$                ;IF NO THEN DO TEST
131650  005737  001006                          TST       @#$PASS            ;IS THIS FIRST PASS
131654  001044                                  BNE       TS662              ;IF NO THEN SHIP TO NEXT TEST
131656                                  70$:
131656  012767  000340  046112                  MOV       #340,PS            ;WE WILL BE PLAYING WITH BIT6
                                                                             ;SO LOCK OUT EXTRAINEOUS INTERRUPTS
131664  012700  131576                          MOV       #DADTBL,R0         ;GET LOCATION OF FIRST REGISTER ADDRESS
131670  012703  131576                  1$:     MOV       #DADTBL,R3         ;MAKE R3 POINT TO LOCATION OF FIRST
                                                                             ;REGISTER ADDRESS
131674  012701  000005                          MOV       #5,R1              ;SET LOOP COUNTER TO CLEAR ALL REG.
131700  005033                          2$:     CLR       @(R3)+             ;CLEAR A REGISTER
131702  077102                                  SOB       R1,2$              ;LOOP UNTIL ALL REGISTERS CLEARED
131704  012770  000100  000000                  MOV       #BIT6,@(R0)        ;SET TEST BIT IN DEVICE REGISTERS
131712  012701  131576                          MOV       #DADTBL,R1         ;GET LOCATION OF FIRST REGISTER ADDRESS
131716  012702  000005                          MOV       #5,R2              ;SET UP TEST LOOP COUNTER
131722  032731  000100                  3$:     BIT       #BIT6,@(R1)+       ;IS TEST BIT SET IN THIS REGISTER
131726  001006                                  BNE       5$                 ;IF YES GO SEE IF THERE IS AN ERROR
131730  077204                          4$:     SOB       R2,3$              ;LOOP UNTIL ALL REGISTER CHECKED
131732  005030                                  CLR       @(R0)+             ;CLEAR REGISTER JUST TESTED AND POINT
                                                                             ;TO NEXT ONE
131734  020027  131606                          CMP       R0,#TBLEND              ;ARE WE DONE TESTING
131740  001407                                  BEQ       7$                 ;IF YES GO TO NEXT TEST
131742  000752                                  BR        1$                 ;CONTINUE TESTING
131744  021041                          5$:     CMP       (R0),-(R1)         ;DID WE COMPARE THE REGISTER TO ITSELF?
131746  001401                                  BEQ       6$
131750  104000                                  EMT                          ;WRITE TO 1 INTERNAL ADDRESS MODIFIED
                                                                             ;ANOTHER SO ADDRESS NOT UNIQUE
131752  062701  000002                  6$:     ADD       #2,R1              ;RESTORE POINTER
131756  000764                                  BR        4$                 ;GET BACK IN TEST LOOP
131760  005000                          7$:     CLR       R0                 ;INITIALIZE LOOP COUNTER
131762  005200                          8$:     INC       R0                 ;INCREMEMT COUNTER
131764  001376                                  BNE       8$                 ;  UNTIL LOOP COUNTER = 0

```

```


                                            ;*******************************************************************************
                                            ;TEST 662        TEST ALL INTERNAL OPTIONS SIMULTANEOUSLY
                                            ;*******************************************************************************
131766                                      TS662:
131766  032737  000001  001020                      BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
131774  001405                                      BEQ     70$             ;IF NO DO TEST
131776  005737  001006                              TST     @#$PASS         ;IS THIS FIRST PASS
132002  001402                                      BEQ     70$             ;IF YES DO TEST
132004  000167  001134                              JMP     ENDPAS          ;IF NO THEN SKIP THIS TEST
132010  000005                              70$:    RESET                   ;CLEAR EVERY BODY
132012  012767  000340  045756                      MOV     #340,PS         ;SET PROCESSOR PRIORITY TO 7
132020  017767  174720  105332                      MOV     @TVECT2,$TMP0
132026  017767  174706  105326                      MOV     @RVECT2,$TMP1
132034  017767  172604  105322                      MOV     @TVECT,$TMP2
132042  017767  172572  105316                      MOV     @RVECT,$TMP3
132050  017767  172576  105312                      MOV     @RTCVT,$TMP4
132056  005067  000360                              CLR     XMTCT2
132062  005067  000356                              CLR     RECCT2
132066  005067  000354                              CLR     TICKS
132072  012777  132210  174644                      MOV     #XMIT2, @TVECT2 ;SET UP SLU2 TRANSMIT VECTOR
132100  012777  000340  174640                      MOV     #340, @TPSW2    ;AND PSW
132106  012777  132244  174624                      MOV     #REC2, @RVECT2  ;SET UP SLU2 RECEIVER VECTOR
132114  012777  000340  174620                      MOV     #340, @RPSW2    ;AND PSW
132122  012777  132200  172522                      MOV     #TICKER, @RTCVT ;SET UP RTC VECTOR
132130  012777  000340  172516                      MOV     #340, @RTCPSW   ;AND PSW
132136  052777  000100  174570                      BIS     #BIT6, @TCSR2   ;ENABLE SLU2 XMIT INTERRUPT
132144  052777  000100  174556                      BIS     #BIT6, @RCSR2   ;ENABLE SLU2 RECEIVER INTERRUPT
132152  012703  132544                              MOV     #BUF2, R3       ;SET UP RECEIVER BUFFER
132156  052777  000100  172464                      BIS     #BIT6, @LKS     ;ENABLE RTC INTERRUPTS
132164  012701  177777                      3$:     MOV     #-1, R1         ;INITIALIZE DATA FOR SLU2
132170  005067  045602                              CLR     PS              ;DROP PROCESSOR PRIORITY TO 0
132174  000001                              WAITIO: WAIT                    ;WAIT FOR INTERRUPT
132176  000776                                      BR      WAITIO

132200  005267  000242                      TICKER: INC     TICKS           ;UPDATE COUNT
132204  000167  000056                              JMP     IOHAND          ;GO TO INTERRUPT HANDLER

132210  005267  000226                      XMIT2:  INC     XMTCT2          ;UPDATE XMIT INTERRUPT COUNT
132214  005201                                      INC     R1              ;UPDATE XMIT DATA
132216  010177  174514                              MOV     R1, @TBUF2      ;SEND NEXT CHARACTER
132222  026727  000214  000400                      CMP     XMTCT2, #400    ;IF 256 CHARACTERS HAVE NOT
132230  002403                                      BLT     1$              ;BEEN TRANSFERRED CONTINUE
132232  042777  000100  174474                      BIC     #BIT6, @TCSR2   ;ELSE NO MORE XMIT INTERRUPTS
132240  000167  000022                      1$:     JMP     IOHAND          ;GO TO INTERRUPT HANDLER

132244  005267  000174                      REC2:   INC     RECCT2          ;UPDATE RECEIVER INTERRUPT COUNT
132250  005777  174456                              TST     @RBUF2          ;BIT 15 SETS IF ANY ERRORS OCCURRED
132254  100002                                      BPL     3$              ;IF BIT IS CLEAR NO ERRORS
132256  000005                                      RESET                   ;CLEAR THE WORLD - STOP ALL
                                                                            ;INTERRUPTS
132260  104000                                      EMT                     ;RECEIVER STATUS ERROR
132262  117723  174444                      3$:     MOVB    @RBUF2, (R3)+   ;GET DATA AND STORE IT

132266  026727  000154  000074              IOHAND: CMP     TICKS, #74      ;HAS 1 SEC ELAPSED
```

```
132274  001401                            BEQ     1$              ;IF YES STOP TEST
132276  000002                            RTI                     ;RETURN FROM INTERRUPT TO AWAIT NEXT
132300  042777  000100  172326  1$:       BIC     #BIT6, @TCSR    ;IF YES STOP TRANSMISSIONS
132306  042777  000100  174420            BIC     #BIT6, @TCSR2   ;
132314  042777  000100  172326            BIC     #BIT6,  @LKS    ;TURN OFF LINE CLOCK

132322  106427  000000          WAITER:   MTPS    #0              ;LOWER PRIORITY TO ALLOW TIME FOR RECEIVER TO FINISH
132326  012705  140000                    MOV     #-40000,R5      ;SET UP LOOP COUNTER
132332  062705  000001          1$:       ADD     #1, R5          ;DO LOOP UNTIL R5 = 0
132336  001375                            BNE     1$
132340  000005                            RESET                   ;STOP EVERYONE SHOULD BE DONE


132342  026767  000074  000074  CHECK2:   CMP     XMTCT2, RECCT2  ;#OF XMIT INTERRUPTS = REC INTERRUPTS
132350  001401                            BEQ     1$
132352  104000                            EMT                     ;INTERRUPT COMPARISON ERROR
132354  012703  132544          1$:       MOV     #BUF2, R3       ;INITIALIZE TO FIRST RECEIVED DATA
132360  005001                            CLR     R1              ;INITIALIZE TO FIRST XMIT DATA
132362  016704  000054                    MOV     XMTCT2, R4      ;GET # OF BYTES TRANSFERRED
132366  122301                  2$:       CMPB    (R3)+, R1       ;IS RECEIVED DATA = EXPECTED DATA
132370  001401                            BEQ     3$
132372  104000                            EMT                     ;SLU2 DATA COMPARISON ERROR
132374  005201                  3$:       INC     R1              ;UPDATE TO NEXT GOOD DATA
132376  077405                            SOB     R4,2$           ;LOOP UNTIL ALL DATA CHECKED
132400  016777  104754  174336  FINIE:    MOV     $TMP0, @TVECT2  ;RESTORE VECTORS
132406  016777  104750  174324            MOV     $TMP1, @RVECT2
132414  016777  104744  172222            MOV     $TMP2, @TVECT
132422  016777  104740  172210            MOV     $TMP3, @RVECT
132430  016777  104734  172214            MOV     $TMP4, @RTCVT
132436  000167  000502                    JMP     ENDPAS          ;FINISHED TESTING GO TO END OF PASS

132442  000000                  XMTCT2:   .WORD   0
132444  000000                  RECCT2:   .WORD   0
132446  000000                  TICKS:    .WORD   0

132450  012737  000007  001002  ERROR8:   MOV     #7,@#$FATAL                       ;SET UP FATAL ERROR NUMBER
132456  012767  000001  046314            MOV     #1,$MSGTY                         ;SET FATAL ERROR FLAG
132464  032737  000001  001020            BIT     #1,@#$ENV                         ;UNDER APT ?
132472  001004                            BNE     COMHLT                            ;YES
132474  012700  132506                    MOV     #COMMSG,R0
132500  004767  000624                    JSR     PC,TYPE
132504  000777                  COMHLT:   BR      .

132506  040506  046111  051125  COMMSG:   .ASCIZ  /FAILURE DURING COMMON TEST/<12><15>
132514  020105  052504  044522
132522  043516  041440  046517
132530  047515  020116  042524
132536  052123  006412     000
        132544                            .EVEN

132544  000200                  BUF2:     .BLKW   200




133144  005327                  ENDPAS:   DEC     (PC)+           ;DECREMENT TEST LOOP COUNTER
```

```
133146  000001                   $EOPCT: .WORD   1
133150  003043                           BGT     $DOAGN           ;IF COUNTER NOT 0 DO TEST AGAIN
133152  005267  045630                   INC     $PASS            ;INCREMENT PASS COUNTER
133156  042767  100000  045622           BIC     #100000,$PASS    ;DON'T LET IT BE NEGATIVE
133164  016767  045634  177754           MOV     $USWR,$EOPCT     ;RESET TEST LOOP COUNTER
133172  012700  133403                   MOV     #ENDMSG,R0       ;LET R0 POINT TO ENDPASS MESSAGE
133176  004767  000126                   JSR     PC,TYPE          ;GO TYPE END PASS MESSAGE
133202  016700  044634                   MOV     42,R0            ;GET MONITOR ADDRESS
133206  001405                           BEQ     DOAGIN           ;IF = 0 NO MONITOR SO DON'T STOP
133210  000005                           RESET                    ;IF MONITOR CLEAR THE WORLD
133212  004710                   $ENDAD: JSR     PC,(R0)          ;GO TO MONITOR
133214  000240                           NOP                      ;THESE THREE LOCATIONS RESERVED
133216  000240                           NOP
133220  000240                           NOP
133222  013737  000004  037360   DOAGIN: MOV     @#4,@#$TMP0
133230  012737  133246  000004           MOV     #1$,@#4
133236  012737  000001  164000           MOV     #1,@#164000
133244  000402                           BR      2$
133246  062706  000004           1$:     ADD     #4,SP
133252  013737  037360  000004   2$:     MOV     @#$TMP0,@#4
133260  000137                   $DOAGN: JMP     @(PC)+           ;RETURN TO TEST AT LOCATION RESTRT
133262  001226                           .WORD   RESTRT



                                 ;*****************************************************************
                                 ;*COMMON SUBROUTINES THAT ARE NEEDED BY THE PROGRAM
                                 ;*
                                 ;*****************************************************************

133264  012737  133274  000024   PWRDN:  MOV     #PWRUP,@#24      ;SET UP POWER FAIL VECTOR FOR POWER UP
133272  000000                           HALT

133274  012737  133264  000024   PWRUP:  MOV     #PWRDN,@#24      ;SET UP POWER FAIL VECTOR FOR POWER DOWN
133302  012706  001000                   MOV     #STBOT,SP        ;SET UP STACK
133306  005737  000172                   TST     @#MTFLAG         ;ARE WE ON MULTI-OPTION TESTER
133312  001004                           BNE     1$               ;IF YES SKIP TYPE OUT
133314  012700  133364                   MOV     #PWRMSG,R0       ;POINT R0 TO POWER FAIL MESSAGE
133320  004767  000004                   JSR     PC,TYPE          ;GO TYPE IT
133324  000167  045676           1$:     JMP     RESTRT           ;GO RESTART TEST


133330  132767  000040  045463   TYPE:   BITB    #40,$ENVM        ;TYPE OUTS DISABLED
133336  001011                           BNE     3$               ;IF YES GET TO EXIT
133340  105737  177564           1$:     TSTB    @#TTCSR          ;TEST FOR PRINTER READY BIT
133344  100375                           BPL     1$               ;IF NOT READY WAIT FOR IT
133346  112037  177566                   MOVB    (R0)+,@#TPB      ;WHEN READY PRINT A CHARACTER
133352  001372                           BNE     1$               ;IF LAST CHARACTER NOT NULL CONTINUE TYPING
133354  105737  177564           2$:     TSTB    @#TTCSR          ;WAIT FOR PRINTER TO FINISH
133360  100375                           BPL     2$
133362  000207                   3$:     RTS     PC

                                 ;*****************************************************************
                                 ;*MESSAGES
                                 ;*
                                 ;*****************************************************************
```

```

133364  047520  042527  020122  PWRMSG: .ASCIZ  /POWER FAILED/<12><15>
133372  040506  046111  042105
133400  006412     000
133403     105  042116  047440  ENDMSG: .ASCIZ  /END OF PASS CJKDJBO/<12><15>
133410  020106  040520  051523
133416  041440  045512  045104
133424  030102  006412     000
133431     012  051415  040524  STRMSG: .ASCIZ  <12><15>/START TESTING/<12><15>
133436  052122  052040  051505
133444  044524  043516  006412
133452     000


        000001                          .END
```

A	020450	AC1	=%000001	AMSGLG=	000000	BBBP1	074466	BDONE	055022
AAADON	074232	AC2	=%000002	AMSGTY=	000000	BBBP2	074476	BDVHLT	054472
AAA1	073432	AC3	=%000003	AMTYP1=	000000	BBB0	074236	BDVMSG	054474
AAA10	073762	AC4	=%000004	AMTYP2=	000000	BBB1	074264	BDVTST	053266
AAA11	074012	AC5	=%000005	AMTYP3=	000000	BBB2	074316	BDVTS2	054030
AAA12	074042	AC6	=%000006	AMTYP4=	000000	BBB3	074336	BELL =	000240
AAA13	074072	AC7	=%000007	ANSR	053754	BBB4	074370	BIC1	013656
AAA2	073462	ADC1	014414	APASS =	000000	BBB5	074416	BIC2	013660
AAA3	073512	ADC2	014416	APRIOR=	000000	BBB6	074424	BIC3	013676
AAA4	073542	ADC3	014436	APTENV=	000001	BBB7	074304	BIS1	013722
AAA5	073572	ADC4	014440	AROUN	001562	BBCDON	115146	BIS2	013724
AAA6	073622	ADC5	014456	AROUND	025044	BBCTP1	115140	BIS3	013744
AAA7	073652	ADDW0 =	000000	ASLB1	015774	BBC10	115144	BITCHK	020124
AAA8	073702	ADDW1 =	000000	ASLB2	015762	BBC2	115114	BITCLR	020052
AAA9	073732	ADDW10=	000000	ASLB3	015764	BBDATO	067274	BITCON	020200
AABBFO	110230	ADDW11=	000000	ASLB4	016006	BBDDON	124444	BITSET	020070
AABDON	110250	ADDW12=	000000	ASL1	015366	BBDONE	067424	BITST1	013612
AABTP1	110240	ADDW13=	000000	ASL2	015370	BBD1	124224	BITST2	013614
AAB2	110226	ADDW14=	000000	ASL3	015406	BBPATO	067304	BITST3	013632
AACDON	115072	ADDW15=	000000	ASL4	015410	BBPAT1	067314	BIT0 =	000001
AACTP1	115064	ADDW2 =	000000	ASL5	015424	BBPAT2	067324	BIT1 =	000002
AAC10	115070	ADDW3 =	000000	ASL6	015426	BBPAT3	067334	BIT10 =	002000
AAC2	115040	ADDW4 =	000000	ASL7	015452	BBPAT4	067344	BIT11 =	004000
AADATO	065714	ADDW5 =	000000	ASRB1	016032	BBPAT5	067354	BIT12 =	010000
AADBF	124210	ADDW6 =	000000	ASRB2	016044	BBPAT6	067364	BIT13 =	020000
AADDON	124214	ADDW7 =	000000	ASRB3	016046	BBP10	067404	BIT14 =	040000
AADONE	066014	ADDW8 =	000000	ASRB4	016054	BBP11	067414	BIT15 =	100000
AAD1	124134	ADDW9 =	000000	ASRB5	016056	BBP7	067374	BIT2 =	000004
AAPATO	065724	ADD1	014276	ASRB6	016076	BB10	066752	BIT3 =	000010
AAPAT1	065734	ADD2	014300	ASRB7	016100	BB13	066760	BIT4 =	000020
AAPAT2	065744	ADD3	014314	ASR1	015476	BB14	066770	BIT5 =	000040
AAPAT3	065754	ADD4	014316	ASR2	015500	BB15	067010	BIT6 =	000100
AAPAT4	065764	ADD5	014334	ASR3	015522	BB16	067034	BIT7 =	000200
AAPAT5	065774	ADD6	014336	ASR4	015524	BB17	067042	BIT8 =	000400
AAPAT6	066004	ADD7	014350	ASR5	015540	BB2	066650	BIT9 =	001000
AA11	065604	ADD8	014352	ASR6	015542	BB20	067052	BRCT	001606
AA12	065606	ADD9	014372	ASR7	015572	BB21	067072	BRH	001572
AA13	065624	ADEVCT=	000000	ASTART	025534	BB22	067116	BRTAB	021242
AA2	065534	ADEVM =	000000	ASWREG=	000000	BB25	067124	BR1	021442
AA20	065632	ADONE	054766	ATESTN=	000000	BB26	067134	BR10	021604
AA22	065656	AENV =	000000	ATRAP	023460	BB27	067154	BR11	021664
AA23	065660	AENVM =	000000	AUNIT =	000000	BB3	066654	BR12	021724
AA24	065700	AERR1	054756	AUSWR =	000000	BB30	067176	BR13	021764
AA27	065706	AFATAL=	000000	AUTO1	023552	BB31	067204	BR14	022024
AA3	065536	AMADR1=	000000	AVECT1=	000000	BB32	067214	BR15	022076
AA4	065554	AMADR2=	000000	AVECT2=	000000	BB33	067234	BR16	022110
AA7	065562	AMADR3=	000000	A1	054726	BB34	067256	BR17	022122
ABASE =	000000	AMADR4=	000000	A11	054726	BB35	067264	BR2	021462
ACDW1 =	000000	AMAMS1=	000000	A12	054742	BB4	066672	BR20	022134
ACDW2 =	000000	AMAMS2=	000000	A2	054760	BB5	066700	BR21	022146
ACPUOP=	000000	AMAMS3=	000000	BA	051536	BB6	066710	BR22	022160
ACTSUM	053762	AMAMS4=	000000	BBBDON	074506	BB7	066730	BR23	022172
ACO	=%000000	AMSGAD=	000000	BBBER1	074302	BCF	053750	BR3	021476

BR4	021514	CCX20	066262	CPUHLT	002164	DDD7	075544	DIVDT	075722
BR5	021532	CCX21	066306	CPUMSG	002166	DDONE	055420	DIVFSU	075122
BR6	021550	CCX24	066314	CSRMSG	051451	DDP0	070204	DIVFT	075230
BR7	021566	CCX25	066324	CSR1	051532	DDP1	070214	DLOOP	054072
BR70	023300	CCX26	066344	CSR2	051534	DDP2	070224	DL11W	023620
BR71	024014	CCX27	066366	CTRAP	023476	DDP3	070234	DNMB0A	007472
BTCON	020236	CCX3	066062	C15	055032	DDP4	070244	DNMB0B	007474
BTERR	020230	CCX30	066374	C2	055046	DDP5	070254	DNMB2A	007626
BTRAP	023526	CCX31	066404	C25	055054	DDP6	070264	DNMB2B	007630
BUF2	132544	CCX32	066424	C3	055072	DDP7	070274	DNMB2C	007636
B1	054776	CCX33	066446	C35	055100	DDP8	070304	DNMB2D	007652
B2	055000	CCX36	066454	C4	055116	DDP9	070314	DNMB2E	007654
B3	055020	CCX6	066070	C45	055122	DD10	067556	DNMB2F	007664
CARRY1	003266	CCX7	066100	C5	055140	DD11	067564	DNMB3A	007730
CC =	177776	CCX8	066120	C55	055144	DD12	067600	DNMB3B	007732
CCBDON	110310	CCX9	066142	C6	055160	DD13	067620	DNMB3C	007742
CCB10	110306	CC1	001566	C65	055166	DD14	067642	DNMB3D	007760
CCB2	110262	CC2	020332	C7	055204	DD15	067650	DNMB3E	007762
CCCDON	075240	CC3	020316	C75	055212	DD16	067664	DNMB4A	010102
CCC1	074512	CDONE	055252	C8	055230	DD17	067704	DNMB4B	010104
CCC10	074776	CHECK2	132342	C85	055234	DD18	067726	DNMB4C	010114
CCC11	075022	CHKAPT	050410	DADTBL	131576	DD2	067450	DNMB4D	010124
CCC12	075046	CHKSUM	053764	DAERR	020176	DD21	067734	DNMB4E	010126
CCC13	075072	CHKWRD	054412	DALTB1	044126	DD22	067744	DNMB4F	010134
CCC2	074536	CIS	025332	DALTB2	044162	DD23	067764	DNM03A	007072
CCC3	074562	CISADR	025036	DALTB3	044412	DD24	070006	DNM03B	007074
CCC4	074606	CISTRP	025022	DALTB4	044446	DD27	070014	DNM03C	007104
CCC5	074632	CLRCD	020314	DATA	051542	DD3	067472	DNM1	007004
CCC6	074656	CLR1	014152	DBE	024644	DD30	070024	DNM1A	007524
CCC7	074702	CMPREG	050226	DBE1	024652	DD31	070044	DNM1B	007526
CCC8	074726	CMPSUB	074126	DBE2	024650	DD32	070066	DNM2	007012
CCC9	074752	CMPTMP	074222	DBE3	024664	DD35	070074	DNM2A	007556
CCDATO	066464	CMP1	014574	DBE4	024676	DD36	070110	DNM2B	007560
CCERR	020422	CMP2	014576	DBE5	024706	DD37	070130	DNM2C	007566
CCP0	066474	CMP3	014620	DDBBF0	110474	DD38	070152	DNM2D	007570
CCP1	066504	CMP4	014622	DDBDON	110504	DD41	070160	DNM3	007022
CCP10	066574	CMP5	014646	DDBTP1	110454	DD6	067500	DNM4	007036
CCP11	066604	CMP6	014650	DDBTP2	110464	DD7	067514	DNM4A	010026
CCP12	066614	CMP7	014670	DDB2	110410	DD8	067534	DNM4B	010030
CCP2	066514	CNTR	050534	DDB5	110452	DEC1	014052	DNM4C	010036
CCP3	066524	COMHLT	132504	DDCDON	115246	DEC2	014054	DNM5A	010176
CCP4	066534	COMMSG	132506	DDCTP1	115232	DEC3	014070	DNM5B	010200
CCP5	066544	COMPLE	054406	DDC10	115244	DEC4	014072	DNM5C	010210
CCP6	066554	COM1	014712	DDC2	115200	DEC5	014106	DNM6A	010252
CCP7	066564	CONCIS	025076	DDDATO	070174	DEC6	014110	DNM6B	010254
CCXDON	066624	CONT	001610	DDDDON	075732	DEC7	014132	DNM6C	010264
CCX12	066150	CONTIN	001220	DDDONE	070324	DERR1	055354	DNM7A	010330
CCX13	066160	CONT42	131334	DDD1	075244	DERR2	055322	DNM7B	010332
CCX14	066200	CON1	020340	DDD2	075304	DEVADR	051530	DNM7C	010342
CCX15	066224	CON2	020424	DDD3	075344	DEVECT	051524	DOAGIN	133222
CCX18	066232	CORH	023520	DDD4	075404	DEV1	051526	DOPB2A	007312
CCX19	066242	COUNT	025412	DDD5	075444	DISPRE	000174	DOPB2B	007344
CCX2	066040	COUNTR	053752	DDD6	075504	DIVDSU	075610	DOP0A	006640

DOP0B	006656	EEP3	070542	ESR29	052310	FDAT00	055764	FXDAT4	056016
DOP0C	006670	EEP4	070552	ESR3	050576	FDAT01	055766	FXDAT5	056020
DOP0D	006712	EERR0	055464	ESR30	052346	FDAT02	055770	FXDAT6	056022
DOP03A	006752	EE12	070464	ESR31	052344	FDAT03	055772	FXDAT7	056024
DOP03B	006754	EE2	070350	ESR32	052566	FDAT04	055774	F10	055654
DOP1	007212	EE3	070372	ESR33	052404	FDAT05	055776	F11	055656
DOP2	007262	EE6	070400	ESR34	051374	FDAT06	056000	F12	055672
DOP4	011324	EE7	070414	ESR35	052460	FDAT07	056002	F13	055716
DOP5	011372	EE8	070434	ESR36	126260	FDONE	056026	F135	055702
DPAT3	062146	EE9	070456	ESR37	126276	FERR20	055740	F14	055724
DRLP	054410	EISEND	037340	ESR38	126314	FFBBF1	110756	F2	055514
DTRAP1	023516	EMTA =	104377	ESR39	126310	FFBDON	110770	F22	055726
DUMMY =	000000	ENDMSG	133403	ESR4	051062	FFBTP1	110746	F3	055534
D1	055276	ENDPAS	133144	ESR40	053370	FFB10	110766	F4	055536
D2	055310	ENT176	031222	ESR41	053412	FFB2	110706	F5	055552
D3	055312	ENT51	027146	ESR42	053434	FFCDON	115454	F6	055620
D4	055316	ER	001604	ESR43	053454	FFCTP1	115432	F7	055634
D5	055324	ERRORA	051332	ESR44	053474	FFCTP2	115442	GAND0	060414
D6	055340	ERRORB	053160	ESR45	053516	FFC10	115452	GAND1	060416
D7	055350	ERRORC	054436	ESR46	053536	FFC2	115406	GAND2	060420
D8	055404	ERRORD	054532	ESR47	053556	FFDAT0	070726	GAND3	060422
D9	055416	ERRORE	051520	ESR48	053604	FFDONE	071006	GCMP	060344
EDONE	055466	ERROR1	002130	ESR49	053616	FFFDON	077016	GDAT00	060434
EEBBF0	110620	ERROR2	002466	ESR5	051064	FFF1	076470	GDAT01	060436
EEBBF1	110630	ERROR3	050434	ESR50	053646	FFF2	076530	GDAT02	060440
EEBDON	110642	ERROR4	124476	ESR51	126274	FFF3	076570	GDAT03	060442
EEBTP1	110610	ERROR5	125720	ESR52	126326	FFF4	076630	GDONE	060444
EEB10	110640	ERROR6	126640	ESR53	126340	FFP0	070736	GFLAG1	060370
EEB2	110550	ERROR7	131504	ESR54	126362	FFP1	070746	GFLAG2	060372
EECDON	115354	ERROR8	132450	ESR55	126366	FFP2	070756	GGBBF0	111104
EECTP1	115330	ERRVEC=	000004	ESR57	051122	FFP3	070766	GGBBF1	111114
EECTP2	115340	ESR0	050714	ESR6	051176	FFP4	070776	GGBDON	111126
EEC10	115352	ESR1	050656	ESR7	051200	FF10	070716	GGBTP1	111074
EEC2	115276	ESR10	053342	ESR8	051254	FF2	070606	GGB10	111124
EEDAT0	070500	ESR11	051640	ESR9	051234	FF3	070630	GGB2	111034
EEDONE	070562	ESR12	052674	ESR98	052034	FF4	070636	GGCDON	115550
EEEDON	076464	ESR13	052706	ESR99	052006	FF5	070646	GGCTP1	115534
EEE1	075736	ESR14	052710	EXADHT	051404	FF6	070666	GGC10	115546
EEE10	076222	ESR15	053046	EXATST	050540	FF7	070710	GGC2	115502
EEE11	076246	ESR16	053124	EXPSUM	053760	FINIE	132400	GGDAT0	072444
EEE12	076272	ESR17	053026	EXTMSG	051406	FINISH	025376	GGDONE	072564
EEE13	076316	ESR18	053102	E1	055432	FIRST =	%000005	GGER0	071712
EEE2	075762	ESR19	051264	E3	055440	FOVER	022720	GGER14	072350
EEE3	076006	ESR2	050726	E4	055442	FPEXIT	124472	GGGDON	101602
EEE4	076032	ESR20	052606	F =	000063	FPHLT	124532	GGG1	100652
EEE5	076056	ESR21	052114	FDATI0	055742	FPMSG	124534	GGG10	101202
EEE6	076102	ESR22	052150	FDATI1	055744	FPP	025366	GGG11	101232
EEE7	076126	ESR23	052152	FDATI2	055746	FPSTRT	054642	GGG12	101262
EEE8	076152	ESR24	052214	FDATI3	055750	FPVECT=	000244	GGG13	101312
EEE9	076176	ESR25	052546	FDATI4	055752	FXDAT0	056006	GGG14	101342
EEP0	070510	ESR26	052534	FDATI5	055754	FXDAT1	056010	GGG2	100702
EEP1	070522	ESR27	052250	FDATI6	055756	FXDAT2	056012	GGG3	100732
EEP2	070532	ESR28	052252	FDATI7	055760	FXDAT3	056014	GGG4	100762

GGG5	101012	GPAT12	060410	HCLR1	061130	HH12	071176	HX4	072642
GGG6	101042	GPAT13	060412	HCMP	061064	HH13	071220	HX7	072650
GGG7	101072	GRESET	060324	HCMP1	061104	HH16	071226	HX8	072664
GGG8	101122	GSETUP	060246	HCMP2	061112	HH17	071242	HX9	072706
GGG9	101152	GS1	060276	HDAT1	061262	HH18	071262	H1	060450
GGP1	072454	G1	056662	HDAT2	061272	HH19	071304	H10	060712
GGP2	072464	G10	057114	HDAT3	061302	HH2	071032	H11	060744
GGP3	072474	G11	057116	HDAT4	061312	HH20	071312	H12	060776
GGP4	072504	G12	057162	HDAT5	061322	HH21	071326	H2	060470
GGP5	072514	G13	057214	HDONE	061332	HH22	071346	H3	060500
GGP6	072524	G14	057216	HERE	= 000000	HH23	071370	H4	060552
GGP7	072534	G15	057262	HFLAG	061140	HH24	071376	H5	060574
GGP8	072544	G16	057314	HHBBF0	111252	HH3	071054	H6	060626
GGP9	072554	G17	057316	HHBBF1	111262	HH6	071062	H7	060660
GG10	072006	G2	056714	HHBDON	111274	HH7	071072	IDATI0	056320
GG11	072034	G20	057362	HHBTP1	111232	HH8	071112	IDATI1	056322
GG12	072056	G21	057414	HHB10	111272	HH9	071134	IDATI2	056324
GG13	072064	G22	057416	HHB2	111172	HICORE	023412	IDATI3	056326
GG14	072100	G23	057462	HHCDON	115660	HLT	= 000000	IDATO0	056310
GG15	072126	G24	057514	HHCTP1	115636	HSTD	061006	IDATO1	056312
GG16	072132	G25	057516	HHCTP2	115646	HXDAT0	073316	IDATO2	056314
GG17	072166	G26	057562	HHC10	115656	HXDONE	073426	IDATO3	056316
GG18	072174	G27	057614	HHC2	115604	HXER9	073022	IDONE	056330
GG19	072224	G3	056716	HHDAT0	071412	HXP1	073326	IIBBF0	111420
GG2	071614	G30	057616	HHDONE	071562	HXP2	073336	IIBBF1	111430
GG20	072252	G31	057662	HHHDON	103016	HXP3	073346	IIBDON	111442
GG21	072274	G32	057714	HHH1	101606	HXP4	073356	IIBTP1	111400
GG22	072302	G33	057716	HHH10	102356	HXP5	073366	IIB10	111440
GG23	072316	G34	057762	HHH11	102426	HXP6	073376	IIB2	111340
GG24	072344	G35	060014	HHH12	102476	HXP7	073406	IICDON	115724
GG25	072352	G36	060016	HHH13	102546	HXP8	073416	IIC2	115702
GG26	072406	G37	060062	HHH2	101656	HX10	072720	IIC20	115722
GG27	072414	G4	056762	HHH3	101726	HX11	072742	IIIDON	077320
GG3	071636	G40	060114	HHH4	101776	HX14	072750	III1	077022
GG4	071644	G41	060116	HHH5	102046	HX15	072764	III2	077052
GG5	071660	G42	060162	HHH6	102116	HX16	073002	III3	077102
GG6	071706	G43	060214	HHH7	102166	HX165	073006	III4	077132
GG7	071714	G44	060216	HHH8	102236	HX17	073024	IJMP	013074
GG8	071750	G5	057014	HHH9	102306	HX18	073044	IJMP4	012712
GG9	071756	G6	057016	HHP0	071422	HX19	073064	IJMP5	013044
GIN1	025110	G7	057062	HHP1	071432	HX2	072610	ILLA	= 004700
GIN2	025152	HADR	061136	HHP10	071542	HX20	073072	ILLB	= 000100
GIN3	025200	HA1R	061212	HHP11	071552	HX22	073114	INC1	013766
GOR0	060424	HA1W	061142	HHP2	071442	HX23	073134	INC2	013770
GOR1	060426	HA2R	061222	HHP3	071452	HX26	073142	INC3	014012
GOR2	060430	HA2W	061152	HHP4	071462	HX28	073166	INC4	014014
GOR3	060432	HA3R	061232	HHP5	071472	HX29	073210	INC5	014030
GPAT00	060374	HA3W	061162	HHP6	071502	HX3	072622	INST	025400
GPAT01	060376	HA4R	061242	HHP7	071512	HX30	073216	IOHAND	132266
GPAT02	060400	HA4W	061172	HHP8	071522	HX31	073232	IPAT10	056270
GPAT03	060402	HA5R	061252	HHP9	071532	HX32	073252	IPAT11	056272
GPAT10	060404	HA5W	061202	HH10	071142	HX33	073274	IPAT12	056274
GPAT11	060406	HCLR	061120	HH11	071156	HX34	073302	IPAT13	056276

IPAT20	056300	JMP3	012656	KDAT00	062130	KKK5	100016	LLC5	117216
IPAT21	056302	JMP3A	012666	KDAT01	062132	KPAT0	062140	LLC6	117254
IPAT22	056304	JMP3B	012700	KDAT02	062134	KPAT1	062142	LLC7	117312
IPAT23	056306	JMP4	012736	KDAT03	062136	KPAT2	062144	LLLDON	100646
ITRAP5=	000004	JMP4A	012746	KDONE	062150	KSP	=%000006	LLL1	100240
I1	056032	JMP4B	012760	KERSTK=	001000	KWSTRT	126002	LLL2	100304
I105	056132	JMP5	013020	KIPAR0=	172340	KX2	062026	LLL3	100350
I12	056134	JMP5A	013032	KIPAR1=	172342	KX3	062030	LLL4	100414
I13	056150	JMP6	012772	KIPAR2=	172344	KX4	062032	LPAT10	062302
I14	056176	JMP6A	013004	KIPAR3=	172346	KX5	062066	LPAT11	062304
I15	056200	JMP7	013046	KIPAR4=	172350	KX6	062074	LPAT12	062306
I16	056202	JMP7A	013060	KIPAR5=	172352	KX7	062106	LPAT13	062310
I17	056216	JSRCK	013460	KIPAR6=	172354	K1	021360	LPAT20	062312
I2	056050	JSRCKA	013454	KIPAR7=	172356	K10	021376	LPAT21	062314
I20	056256	JSRCK1	013476	KIPDR0=	172300	K11	021400	LPAT22	062316
I23	056264	JSRSEQ	013456	KIPDR1=	172302	K12	021402	LPAT23	062320
I3	056074	JSR0	013114	KIPDR2=	172304	K2	021362	LSREG =	177524
I4	056076	JSR1	013120	KIPDR3=	172306	K3	021364	LTCHLT	126674
I5	056100	JSR1A	013142	KIPDR4=	172310	K4	021366	LTCMSG	126676
I6	056124	JSR2	013216	KIPDR5=	172312	K5	021370	L1	062154
JBUF0	061744	JSR2A	013246	KIPDR6=	172314	K6	021372	L2	062214
JBUF1	061746	JSR2B	013250	KIPDR7=	172316	K7	021374	L3	062216
JBUF2	061750	JSR3	013144	KKBBF0	111722	LAST	=%000001	L4	062220
JBUF3	061752	JSR3A	013204	KKBBF1	111732	LATCNT	051544	L5	062270
JDATI0	061754	JSR3B	013206	KKBDON	111744	LDATI0	062324	L6	062276
JDATI1	061756	JSR4	013272	KKBTP1	111702	LDATI1	062326	MBDM2A	010464
JDATI2	061760	JSR4A	013310	KKB10	111742	LDATI2	062330	MBDM2B	010466
JDATI3	061762	JSR4B	013312	KKB2	111640	LDATI3	062332	MBDM2C	010476
JDAT00	061764	JSR5	013362	KKCDON	117022	LDAT00	062336	MBDM2D	010510
JDAT0	061774	JSR5A	013406	KKC1	116026	LDAT01	062340	MBDM2E	010512
JDAT01	061766	JSR5B	013410	KKC10	116260	LDAT02	062342	MBDM2F	010522
JDAT02	061770	JSR6	013324	KKC11	116306	LDAT03	062344	MBDM4A	010732
JDAT03	061772	JSR6A	013350	KKC12	116334	LDCDSU	117504	MBDM4B	010742
JDAT1	061776	JSR6AD	013452	KKC13	116362	LDCFSU	116720	MBDM4C	010754
JDAT2	062000	JSR6B	013352	KKC14	116410	LDCT	117012	MBDM4D	010756
JDAT3	062002	JSR7	013422	KKC15	116436	LDONE	062346	MBDM4E	010764
JDONE	062004	JSR7A	013440	KKC16	116464	LDXSUB	120744	MDAT00	062474
JJBBF0	111562	JSR7B	013442	KKC17	116512	LDXT	121072	MDAT01	062476
JJBDON	111574	J1	061644	KKC2	116054	LKS	124650	MDAT02	062500
JJBTP1	111550	J2	061662	KKC20	116540	LKSTST	126034	MDAT03	062502
JJB10	111572	J3	061664	KKC21	116566	LLBBF0	112050	MDM1A	010374
JJB2	111506	J4	061666	KKC22	116614	LLBDON	112062	MDM1B	010376
JJJDON	077702	J5	061722	KKC23	116642	LLBTP1	112040	MDM2A	010424
JJJ1	077324	J6	061730	KKC24	116670	LLB10	112060	MDM2B	010426
JJJ2	077370	J7	061742	KKC3	116102	LLB2	112004	MDM2C	010434
JJJ3	077434	KBUF0	062120	KKC4	116130	LLCDON	117576	MDM2D	010436
JJJ4	077500	KBUF1	062122	KKC5	116156	LLC1	117026	MDM3A	010560
JMPCK	013076	KBUF2	062124	KKC6	116204	LLC10	117350	MDM3B	010562
JMPERR	020262	KBUF3	062126	KKC7	116232	LLC11	117406	MDM3C	010572
JMPSEQ	013110	KDATI0	062110	KKKDON	100234	LLC12	117444	MDM3D	010604
JMPT	020252	KDATI1	062112	KKK1	077706	LLC2	117064	MDM3E	010624
JMP2	012714	KDATI2	062114	KKK3	077736	LLC3	117122	MDM4A	010666
JMP2A	012724	KDATI3	062116	KKK4	077766	LLC4	117160	MDM4B	010670

MDM4C	010676	MMC14	120444	MPAT23	062472	NEG11	004726	NPAT12	062644
MDM5A	011016	MMC15	120512	MRK1	016354	NEG12	004736	NPAT13	062646
MDM5B	011020	MMC16	120560	MRK2	016376	NEG13	004752	NPAT20	062626
MDM5C	011030	MMC17	120626	MRK3	016400	NEG14	004754	NPAT21	062630
MDM5D	011040	MMC2	117650	MRK4	016422	NEG2	014504	NPAT22	062632
MDM5E	011060	MMC20	120674	MRK5	016426	NEG20	005004	NPAT23	062634
MDM6A	011112	MMC3	117716	MRK6	016442	NEG21	005006	N1	062510
MDM6B	011114	MMC4	117764	MTFLAG	000172	NEG22	005026	N12	062560
MDM6C	011124	MMC5	120032	MTPI0	020036	NEG3	014526	N13	062574
MDM6D	011136	MMC6	120100	MTPS1	016466	NEG30	005236	N14	062602
MDM6E	011160	MMC7	120146	MTPS1A	016506	NEG31	005240	N2	062534
MDM7A	011216	MMMDON	103456	MTPS2	016536	NEG32	005246	N3	062536
MDM7B	011220	MMM1	103022	MTPS3	016602	NEG33	005264	N4	062540
MDM7C	011230	MMM2	103056	MTPS4	016644	NEG34	005272	ODATI0	063024
MDM7D	011242	MMM3	103112	MTPS5	016700	NEG4	014530	ODATI1	063026
MDM7E	011260	MMM4	103146	MTPS6	016744	NEG40	005526	ODATI2	063030
MDONE	062504	MMM5	103202	MTPS7	017010	NEG41	005530	ODATI3	063032
MEM	050532	MMUHLT	050470	MULDSU	076674	NEG42	005536	ODAT00	062762
MERR3	062412	MMUMSG	050472	MULDT	077006	NEG5	014550	ODAT01	062764
MFPI0	017750	MMUTST	037372	MULFSU	076346	NEG50	005574	ODAT02	062766
MFPI0A	017770	MMVEC =	000250	MULFT	076454	NEG51	005576	ODAT03	062770
MFPS1	017040	MODDD0	104142	MVLCNT	051546	NEG52	005604	ODONE	063034
MFPS2A	017112	MODDD1	104152	M1	062352	NEG60	005644	OERR0	062732
MFPS2B	017114	MODDOV	103746	M15	062372	NEG61	005646	OOBDON	112474
MFPS2C	017126	MODDSU	102622	M2	062376	NEG70	005700	OOBTP1	112452
MFPS3A	017166	MODDT0	102776	M3	062400	NEG71	005702	OOBTP2	112462
MFPS3B	017170	MODDT1	103006	M4	062402	NERR0	062600	OOB10	112472
MFPS3C	017202	MODFD0	103436	M5	062436	NEXT	050756	OOB2	112406
MFPS4A	017242	MODFD1	103446	M6	062442	NNBBF0	112332	OOCDON	121326
MFPS4B	017244	MODFOV	103242	M7	062450	NNBDON	112344	OOCTB0	121310
MFPS4C	017256	MODFSU	101376	M8	062414	NNBTP1	112312	OOCTB1	121314
MFPS5A	017316	MODFT0	101552	N =	000307	NNBTP2	112322	OOC10	121324
MFPS5B	017320	MODFT1	101562	NATBF1	114776	NNB10	112342	OOC2	121256
MFPS5C	017332	MODP1	101572	NATINS	114702	NNB2	112240	OOODON	105154
MFPS6A	017374	MOR0	020456	NATSUB	114626	NNCDON	121214	OOO1	105110
MFPS6B	017376	MOR1	020500	NBR	001576	NNCTB0	121176	OOO2	105106
MFPS6C	017410	MOR2	020534	NDATI0	062650	NNCTB1	121202	OOO3	105122
MFPS7A	017452	MOR3	020602	NDATI1	062652	NNC10	121212	OOO4	105146
MFPS7B	017454	MOR4	020614	NDATI2	062654	NNC2	121144	OPAT10	063014
MFPS7C	017466	MOR5	020626	NDATI3	062656	NNNDON	104162	OPAT11	063016
MFPT =	000007	MOR6	020672	NDAT00	062606	NNN1	103462	OPAT12	063020
MMBBF0	112176	MOR7	020704	NDAT01	062610	NNN2	103536	OPAT13	063022
MMBBF1	112206	MOR8	020716	NDAT02	062612	NNN3	103612	OPAT20	063000
MMBDON	112220	MOV1	013546	NDAT03	062614	NNN4	103666	OPAT21	063002
MMBTP1	112156	MOV2	013550	NDONE	062660	NOCMP	054362	OPAT22	063004
MMB10	112216	MOV3	013566	NEGAT	026546	NODL	023606	OPAT23	063006
MMB2	112122	MPAT10	062454	NEG00	004650	NODL2	024234	OPAT24	063010
MMCDON	121102	MPAT11	062456	NEG01	004652	NOP =	000240	OVDNTT	077672
MMC1	117602	MPAT12	062460	NEG02	004660	NOR	023406	OVDTT	100636
MMC10	120214	MPAT13	062462	NEG03	004674	NOSUB	023502	OVFNTT	077310
MMC11	120262	MPAT20	062464	NEG04	004676	NOTEQ	054372	OVFTT	100224
MMC12	120330	MPAT21	062466	NEG1	014502	NPAT10	062640	OVUNDN	077550
MMC13	120376	MPAT22	062470	NEG10	004724	NPAT11	062642	OVUNDT	100464

OVUNFN	077166	PSW	= 177776	Q2	063176	RETR3	024600	ROT3E	012134
OVUNFT	100052	PSWORD	025414	Q22HLT	053214	RET1	025222	ROT4	012172
O1	062664	PUSRM	= 030000	Q22MSG	053216	RET2	025234	ROT5	012236
O12	062734	PWRDN	133264	Q22TST	051552	RET3	025244	ROT6	012270
O13	062750	PWRMSG	133364	Q22TS1	052462	RET4	025242	ROT7	012326
O14	062756	PWRUP	133274	Q22TS2	052570	RFLAG	053756	RPSW	124642
O2	062710	P1	063040	Q22TS3	052734	RITSH	030444	RPSW2	126742
O3	062712	P2	063054	Q3	= 063200	ROL1	015156	RRBDON	113126
O4	062714	P3	= 063056	Q4	063202	ROL2	015160	RRBTP1	113102
PCNO1	020732	P4	063060	Q5	063224	ROL3	015176	RRBTP2	113112
PCN1	024510	P5	063102	RBUF	124632	ROL4	015200	RRBTP3	113122
PCN2	020750	QDATI0	063300	RBUF2	126732	ROL5	015214	RRB10	113124
PCN3	021000	QDATI1	063302	RCSR	124630	ROL6	015216	RRB2	113036
PCN4	021024	QDATI2	063304	RCSR2	126730	ROL7	015240	RRCDON	121706
PCN5	021046	QDATI3	063306	RECCT2	132444	RORB1	015616	RRCTB0	121664
PCR	= 177520	QDATO0	063270	REC2	132244	RORB10	015700	RRCTB1	121670
PDATI0	063136	QDATO1	063272	REGR1	025662	RORB11	015702	RRCTB2	121700
PDATI1	063140	QDATO2	063274	REGR2	026010	RORB12	015714	RRC10	121704
PDATI2	063142	QDATO3	063276	REGR23	030104	RORB13	015726	RRC2	121632
PDATI3	063144	QDONE	063310	REGR3	026136	RORB14	015730	RRRDON	105776
PDATO0	063146	QPAT10	063250	REGR4	026262	RORB2	015630	RRREXP	105766
PDATO1	063150	QPAT11	063252	REGR5	026412	RORB3	015632	RRRTP1	105746
PDATO2	063152	QPAT12	063254	REG01	027724	RORB4	015642	RRRTP2	105756
PDATO3	063154	QPAT13	063256	REG1	003502	RORB5	015644	RRR10	105744
PDONE	063156	QPAT20	063260	REG1A	003530	RORB7	015654	RRR2	105676
PDRTB1	044144	QPAT21	063262	REG1E	003514	ROR1	015264	RRR3	105700
PDRTB2	044176	QPAT22	063264	REG2	003562	ROR2	015266	RRR4	105724
PDRTB3	044430	QPAT23	063266	REG2B	003610	ROR3	015304	RTCPSW	124654
PDRTB4	044462	QQBDON	112766	REG2E	003574	ROR4	015306	RTCVT	124652
PNTR	054434	QQBTP1	112734	REG3	003642	ROR5	015324	RTI1	023220
PPAT10	063126	QQBTP2	112754	REG3A	003670	ROR6	015326	RTI2	023224
PPAT11	063130	QQB10	112764	REG3E	003654	ROR7	015342	RTRAP	= 000010
PPAT12	063132	QQB2	112670	REG4	003722	ROTLP1	053574	RTRAP1=	000034
PPAT13	063134	QQCDON	121562	REG4A	003750	ROTLP2	053636	RTRAP2=	000020
PPBDON	112624	QQCTB0	121540	REG4E	003734	ROTX	012240	RTRAP3=	000030
PPBTP1	112602	QQCTB1	121544	REG45	030260	ROTXAD	012330	RTRAP4=	000014
PPBTP2	112612	QQCTB2	121554	REG5	004002	ROT0A	011544	RTRAP5=	000004
PPB10	112622	QQC10	121560	REG5A	004030	ROT0B	011546	RTS1	013520
PPB2	112536	QQC2	121506	REG5E	004014	ROT0C	011570	RTT1	023110
PPCDON	121440	QQQBF0	105456	REG6	004062	ROT1A	011620	RTT2	023116
PPCTB0	121422	QQQBF1	105472	REG6A	004110	ROT1B	011622	RTT3	023150
PPCTB1	121426	QQQDON	105624	REG6E	004074	ROT1C	011646	RTT4	023160
PPC10	121436	QQQTP1	105506	RESET2	024304	ROT1D	011650	RTT5	023122
PPC2	121370	QQQ10	105516	RESET3	024302	ROT1E	011700	RTT6	023172
PPPBF0	105274	QQQ2	105356	REST	017536	ROT2A	011734	RVECT	124640
PPPBF1	105310	QQQ20	105520	RESTRT	001226	ROT2B	011736	RVECT2	126740
PPPDON	105336	QQQ22	105532	RET	025212	ROT2C	011766	RWREG	= 177522
PPPTP1	105324	QQQ23	105564	RETA	050432	ROT2D	011770	ROTRAP	023432
PPP10	105334	QQQ24	105576	RETAT	022764	ROT2E	012024	R1ERR	003546
PPP2	105174	QQQ25	105622	RETBT	023014	ROT3A	012054	R2ERR	003626
PPP3	105224	QQQ3	105406	RETCT	023050	ROT3B	012056	R3ERR	003706
PPP4	105236	QQQ4	105420	RETR1	024516	ROT3C	012104	R4ERR	003766
PS	= 177776	Q1	063162	RETR2	024554	ROT3D	012106	R5ERR	004046

```
R6      =%000006
R6ERR    004126
R7      =%000007
SADR     061614
SBC1     015050
SBC2     015052
SBC3     015070
SBC4     015072
SBC5     015110
SBC6     015112
SBC7     015132
SB0      012344
SB2      012424
SB4      012506
SB5      012550
SB5A     012542
SB5X     012552
SB5XAD   012554
SB6      012604
SB6X     012606
SB7      012636
SB7X     012640
SB7XAD   012642
SCOPE =  000240
SC3      020402
SC4      020416
SDAT00   061630
SDAT01   061632
SDAT02   061634
SDAT03   061636
SDONE    061640
SERR0    061524
SERR1    061432
SERR10   061542
SERR4    061574
SETBR    001456
SETCC    001476
SETCD    020400
SETREG   050134
SETUP    001440
SET2BR   001546
SFPTBL   053666
SHL      003314
SHLE     003330
SHR      003374
SHRE     003410
SIMG0A   051550
SIXDLR   054312
SKTST2   024322
SLU1ST   124656
SLU2ST   126750
SL1HLT   125754
SL1MSG   125756

SL2HLT   131540
SL2MSG   131542
SNMB0A   006016
SNMB1A   006066
SNMB1B   006070
SNMB1C   006112
SNMB2A   006172
SNMB2B   006174
SNMB2C   006202
SNMB2D   006222
SNMB2E   006224
SNMB3A   006324
SNMB3B   006326
SNMB3C   006344
SNMB3D   006346
SNM0A    005774
SNM1A    006042
SNM2A    006136
SNM2B    006140
SNM3A    006262
SNM3B    006264
SNM4A    006400
SNM4B    006402
SNM5A    006436
SNM5B    006440
SNM6A    006476
SNM6B    006500
SNM7A    006534
SNM7B    006536
SOB1     016270
SOB2     016276
SOB3     016300
SOB4     016320
SOPA     005730
SOPB     005750
SOPB0A   004324
SOPB0B   004334
SOPB1A   004376
SOPB1B   004414
SOPB1C   004434
SOPB1D   004454
SOPB2A   004540
SOPB2B   004560
SOPB2C   004604
SOPB2D   004630
SOPB3A   005114
SOPB3B   005140
SOPB3C   005162
SOPB3D   005204
SOPX     005722
SOPXAD   005752
SOPZA    004476
SOP0A    004240

SOP0B    004252
SOP0C    004270
SOP0D    004314
SOP1A    004346
SOP1B    004360
SOP2B    004516
SOP3A    005054
SOP3B    005070
SOP4A    005324
SOP4B    005344
SOP5A    005376
SOP5B    005412
SOP6A    005432
SOP6B    005446
SOP7A    005470
SOP7B    005504
SPAT10   061620
SPAT11   061622
SPAT12   061624
SPAT13   061626
SR0    = 177572
SR1    = 177574
SR2    = 177576
SR3    = 172516
SSBDON   113270
SSBTP1   113244
SSBTP2   113254
SSBTP3   113264
SSB10    113266
SSB2     113200
SSCDON   122032
SSCTB0   122014
SSCTB1   122020
SSC10    122030
SSC2     121752
SSSA1    106140
SSSBF0   106130
SSSDON   106162
SSSTP1   106150
SSS10    106160
SSS2     106050
START    001200
STATUS=  177776
STBOT    001000
STCDFS   107730
STCDT    110060
STCFDS   107276
STCFT    107426
STCIBF   123414
STCSUB   123266
STP      022722
STP3     023356
STP3D    023360

STP4     024460
STP4E    024462
STRMSG   133431
STXBF    124010
STXSUB   123676
SUB0     006620
SUB0A    006622
SUB1     014736
SUB2     014740
SUB3     014762
SUB4     014764
SUB5     015002
SUB6     015004
SUB7     015024
SWB1     014234
SWB2     014236
SWB3     014254
SWMSG    054570
SWR      021404
SWREG    000176
SXT0     016136
SXT1     016140
SXT2     016170
SX10     061466
SX11     061512
SX12     061520
SX2      061370
SX3      061372
SX4      061376
SX5      061422
SX6      061430
SX7      061434
SX8      061460
SX9      061462
S1       025432
S10      025454
S11      025456
S2       025434
S3       025436
S4       025440
S5       025442
S6       025444
S7       025446
S8       025450
S9       025452
TAB     =%000003
TABLE    025256
TABLES   054432
TBIT   = 000020
TBITPS   037356
TBLEND   131606
TBL1     011336
TBL2     011404

TBUF     124636
TBUF2    126736
TCCBF0   116002
TCCBF1   116012
TCCDON   116022
TCC2     115742
TCC3     115772
TCSR     124634
TCSR2    126734
TDATI0   056624
TDATI1   056626
TDATI2   056630
TDATI3   056632
TDAT00   056614
TDAT01   056616
TDAT02   056620
TDAT03   056622
TDEC1    022222
TDEC2    022236
TDEC3    022266
TDEC4    022320
TDEC6    022334
TDEC7    022336
TDEC77   023702
TDEC8    023700
TDONE    056634
TEMP1    025416
TEMP2    025420
TEMP3    025422
TEMP4    025424
TEMP5    025426
TEMP6    025430
TENSAV   025042
TESTN1   021406
TEST1    014172
TEST2    014174
TEST3    014212
THRPRT   025464
TICKER   132200
TICKS    132446
TKB    = 177562
TMP      050526
TMP1     050530
TOFF     050046
TON      050102
TONT1    023302
T010     021340
T0114    021352
T014     021342
T020     021344
T0244    021354
T0250    021356
T030     021346
```

```
TO34      021350
TO4       021336
TPAT10    056574
TPAT11    056576
TPAT12    056600
TPAT13    056602
TPAT20    056604
TPAT21    056606
TPAT22    056610
TPAT23    056612
TPB     = 177566
TPS     = 177564
TPSW      124646
TPSW2     126746
TRACE     023276
TRAPA   = 000077
TRAPB     023554
TRAPC   = 104777
TRAPPC    037346
TRAPPS    037350
TRAP10    025044
TRCSR   = 177560
TRC1      023346
TRPADR    023430
TRT     = 000003
TR0       024012
TR2       024016
TR3       024124
TR4       024130
TR5       024126
TSTSPC    037306
TST160    030500
TST161    030526
TST162    030544
TST163    030562
TST164    030624
TST165    030662
TST166    030706
TST167    030736
TST170    030774
TST171    031020
TST172    031046
TST173    031070
TST174    031126
TST175    031162
TST176    031234
TST177    031302
TST200    031350
TST201    031422
TST202    031470
TST203    031536
TST204    031610
TST205    031656
TST206    031750
TST207    032026
TST210    032104
TST211    032162
TST212    032236
TST213    032312
TST214    032366
TST215    032444
TST216    032522
TST217    032576
TST220    032652
TST221    032722
TST222    032772
TST223    033042
TST224    033116
TST225    033166
TST226    033236
TST227    033312
TST230    033362
TST231    033432
TST232    033502
TST233    033556
TST234    033626
TST235    033676
TST236    033746
TST237    034022
TST240    034072
TST241    034142
TST242    034232
TST243    034302
TST244    034352
TST245    034422
TST246    034470
TST247    034536
TST250    034604
TST251    034654
TST252    034724
TST253    034772
TST254    035040
TST255    035114
TST256    035170
TST257    035250
TST260    035324
TST261    035400
TST262    035460
TST263    035534
TST264    035610
TST265    035664
TST266    035744
TST267    036020
TST270    036074
TST271    036154
TST272    036230
TST273    036304
TST274    036352
TST275    036430
TST276    036504
TST277    036560
TST300    036634
TST301    036706
TST302    036760
TST303    037032
TST304    037106
TST305    037162
TST306    037234
TST37     026574
TST40     026630
TST41     026644
TST42     026662
TST43     026716
TST44     026750
TST45     027002
TST46     027034
TST47     027072
TST50     027122
TST51     027174
TST52     027234
TST53     027274
TST54     027334
TST55     027372
TST56     027430
TST57     027466
TST60     027526
TST61     027566
TST62     027624
TS1       001440
TS10      002734
TS100     005776
TS101     006020
TS102     006044
TS103     006114
TS104     006150
TS105     006234
TS106     006274
TS107     006356
TS11      003122
TS110     006410
TS111     006450
TS112     006506
TS113     006544
TS114     006562
TS115     006600
TS116     006630
TS117     006724
TS12      003140
TS120     006770
TS121     007046
TS122     007106
TS123     007126
TS124     007146
TS125     007166
TS126     007222
TS127     007244
TS13      003160
TS130     007272
TS131     007322
TS132     007354
TS133     007400
TS134     007424
TS135     007450
TS136     007502
TS137     007534
TS14      003200
TS140     007600
TS141     007674
TS142     007774
TS143     010050
TS144     010144
TS145     010222
TS146     010276
TS147     010354
TS15      003220
TS150     010404
TS151     010446
TS152     010534
TS153     010644
TS154     010706
TS155     010774
TS156     011070
TS157     011172
TS16      003246
TS160     011272
TS161     011340
TS162     011406
TS163     011454
TS164     011522
TS165     011572
TS166     011702
TS167     012026
TS17      003300
TS170     012136
TS171     012174
TS172     012242
TS173     012272
TS174     012332
TS175     012354
TS176     012400
TS177     012434
TS2       001662
TS20      003332
TS200     012460
TS201     012514
TS202     012556
TS203     012610
TS204     012644
TS205     013112
TS206     013500
TS207     013530
TS21      003360
TS210     013570
TS211     013634
TS212     013700
TS213     013746
TS214     014032
TS215     014134
TS216     014154
TS217     014214
TS22      003412
TS220     014256
TS221     014374
TS222     014460
TS223     014552
TS224     014672
TS225     014714
TS226     015026
TS227     015134
TS23      003424
TS230     015242
TS231     015344
TS232     015454
TS233     015574
TS234     015742
TS235     016010
TS236     016110
TS237     016172
TS24      003440
TS240     016260
TS241     016322
TS242     016444
TS243     016510
TS244     016546
TS245     016612
TS246     016652
TS247     016710
TS25      003454
TS250     016754
TS251     017020
TS252     017066
TS253     017136
TS254     017212
TS255     017266
TS256     017342
```

TS257	017420	TS335	023626	TS413	046066	TS472	072570	TS550	113132
TS26	003470	TS336	023726	TS414	046562	TS473	073432	TS551	113274
TS260	017476	TS337	024036	TS415	047314	TS474	074236	TS552	113424
TS261	017536	TS34	003710	TS416	047714	TS475	074512	TS553	113560
TS262	017576	TS340	024170	TS417	047776	TS476	075244	TS554	113722
TS263	017710	TS341	024234	TS42	004130	TS477	075736	TS555	114074
TS264	017770	TS342	024322	TS420	050540	TS5	002342	TS556	115014
TS265	020052	TS343	024476	TS421	050756	TS50	004316	TS557	115076
TS266	020204	TS344	024530	TS422	051104	TS500	076470	TS56	004562
TS267	020242	TS345	024560	TS423	051552	TS501	077022	TS560	115152
TS27	003516	TS346	024610	TS424	052050	TS502	077324	TS561	115252
TS270	020264	TS347	024726	TS425	052462	TS503	077706	TS562	115360
TS271	020456	TS35	003736	TS426	052570	TS504	100240	TS563	115460
TS272	020512	TS350	024764	TS427	052734	TS505	100652	TS564	115554
TS273	020546	TS351	037456	TS43	004152	TS506	101606	TS565	115664
TS274	020636	TS352	037512	TS430	053266	TS507	103022	TS566	115730
TS275	020726	TS353	037554	TS431	053666	TS51	004336	TS567	116026
TS276	020744	TS354	037642	TS432	054700	TS510	103462	TS57	004632
TS277	020766	TS355	037700	TS433	054772	TS511	104166	TS570	117026
TS3	002004	TS356	037724	TS434	055026	TS512	105064	TS571	117602
TS30	003550	TS357	037744	TS435	055256	TS513	105160	TS572	121106
TS300	021016	TS36	003770	TS436	055424	TS514	105342	TS573	121220
TS301	021042	TS360	037764	TS437	055472	TS515	105630	TS574	121332
TS302	021064	TS361	040004	TS44	004172	TS516	106002	TS575	121444
TS303	021102	TS362	040024	TS440	056032	TS517	106166	TS576	121566
TS304	021132	TS363	040142	TS441	056334	TS52	004362	TS577	121712
TS305	021174	TS364	040176	TS442	056640	TS520	106332	TS6	002470
TS306	021424	TS365	040260	TS443	060450	TS521	106476	TS60	004704
TS307	021624	TS366	040350	TS444	061336	TS522	106634	TS600	122036
TS31	003576	TS367	040446	TS445	061644	TS523	107010	TS601	122174
TS310	022064	TS37	004016	TS446	062010	TS524	107442	TS602	122252
TS311	022204	TS370	040552	TS447	062154	TS525	110074	TS603	122330
TS312	022222	TS371	040656	TS45	004212	TS526	110134	TS604	123430
TS313	022246	TS372	041022	TS450	062352	TS527	110254	TS605	123470
TS314	022336	TS373	041206	TS451	062510	TS53	004416	TS606	124026
TS315	022372	TS374	041662	TS452	062664	TS530	110314	TS607	124134
TS316	022426	TS375	042212	TS453	063040	TS531	110354	TS61	004764
TS317	022462	TS376	042560	TS454	063162	TS532	110510	TS610	124224
TS32	003630	TS377	042734	TS455	063314	TS533	110646	TS611	124710
TS320	022516	TS4	002220	TS456	064102	TS534	110774	TS612	124736
TS321	022552	TS40	004050	TS457	064456	TS535	111132	TS613	124764
TS322	022616	TS400	043122	TS46	004232	TS536	111300	TS614	125052
TS323	022652	TS401	043244	TS460	065052	TS537	111446	TS615	125104
TS324	022734	TS402	043522	TS461	065302	TS54	004456	TS616	125132
TS325	022764	TS403	043712	TS462	065514	TS540	111600	TS617	125160
TS326	023024	TS404	044212	TS463	066020	TS541	111750	TS62	005036
TS327	023062	TS405	044476	TS464	066630	TS542	112066	TS620	125276
TS33	003656	TS406	044652	TS465	067430	TS543	112224	TS621	125370
TS330	023116	TS407	045226	TS466	070330	TS544	112350	TS622	125470
TS331	023172	TS41	004076	TS467	070566	TS545	112500	TS623	125606
TS332	023232	TS410	045364	TS47	004260	TS546	112630	TS624	126034
TS333	023316	TS411	045430	TS470	071012	TS547	112772	TS625	126062
TS334	023366	TS412	045562	TS471	071566	TS55	004520	TS626	126216

TS627	126406	TTC2	122104	UPAT12	064032	U2	063422	WPAT02	064436
TS63	005072	TTTA1	106310	UPAT13	064034	U3	063440	WPAT03	064440
TS630	126536	TTTA2	106312	UPAT20	064036	U4	063502	WWBDON	115010
TS631	127002	TTTBF0	106276	UPAT21	064040	U5	063520	WWB1	114074
TS632	127030	TTTDON	106326	UPAT22	064042	U6	063562	WWB10	114506
TS633	127056	TTTTP1	106314	UPAT23	064044	U7	063600	WWB11	114554
TS634	127140	TTT10	106324	UPAT30	064046	VDEC	022364	WWB2	114142
TS635	127210	TTT2	106242	UPAT31	064050	VDEC10	022540	WWB3	114210
TS636	127236	TTT3	106266	UPAT32	064052	VDEC11	022574	WWB4	114256
TS637	127264	TVECT	124644	UPAT33	064054	VDEC12	022600	WWB5	114324
TS64	005142	TVECT2	126744	UPAT40	064056	VDEC13	022640	WWB6	114372
TS640	127376	TYPE	133330	UPAT41	064060	VDEC14	022644	WWB7	114440
TS641	127532	T105	056434	UPAT42	064062	VDEC2	022360	WWCDON	123424
TS642	127666	T12	056436	UPAT43	064064	VDEC3	022420	WWC1	122330
TS643	127760	T13	056452	UROM1	064074	VDEC4	022414	WWC10	122634
TS644	130060	T14	056500	UROM2	064076	VDEC5	022454	WWC11	122670
TS645	130176	T15	056502	UROM3	064100	VDEC6	022450	WWC12	122724
TS646	130322	T16	056504	USESTK=	000600	VDEC7	022510	WWC13	122760
TS647	130406	T17	056522	USP =	%000006	VDEC8	022504	WWC14	123014
TS65	005206	T2	056352	USP1	017552	VDEC9	022544	WWC15	123050
TS650	130434	T20	056562	USP1A	017570	VECT1	051522	WWC16	123104
TS651	130544	T23	056570	USP2	017626	VRTPCR	053746	WWC17	123140
TS652	130646	T3	056376	USP3	017644	VVBDON	110350	WWC2	122364
TS653	130772	T4	056400	USP4	017672	VVB10	110346	WWC20	123174
TS654	131060	T5	056402	USRM =	140000	VVB2	110322	WWC21	123230
TS655	131172	T6	056426	UTMP1	064070	VVCBF0	122316	WWC3	122420
TS656	131252	UDONE	064102	UTMP2	064072	VVCDON	122324	WWC4	122454
TS657	131346	UERR0	063436	UUBDON	113554	VVCTP1	122304	WWC5	122510
TS66	005310	UERR3	063770	UUBTP1	113470	VVC2	122272	WWC6	122544
TS660	131424	UFLAG	064066	UUBTP2	113542	VVVBF0	106606	WWC7	122600
TS661	131640	UIPAR0=	177640	UUB10	113552	VVVDON	106630	WWWBF0	106752
TS662	131766	UIPAR1=	177642	UUB2	113466	VVVTP1	106616	WWWBF1	106772
TS67	005346	UIPAR2=	177644	UUCBF0	122240	VVV10	106626	WWWDON	107004
TS7	002612	UIPAR3=	177646	UUCDON	122246	VVV2	106550	WWWTP1	106762
TS70	005414	UIPAR4=	177650	UUCTP1	122226	WAITER	132322	WWW10	107002
TS71	005450	UIPAR5=	177652	UUC2	122214	WAITIO	132174	WWW2	106714
TS72	005506	UIPAR6=	177654	UUUA1	106454	WASR6	037344	W10	064244
TS73	005544	UIPAR7=	177656	UUUA2	106456	WASSR0	037352	W11	064256
TS74	005614	UIPDR0=	177600	UUUBF0	106442	WASSR2	037354	W12	064276
TS75	005654	UIPDR1=	177602	UUUDON	106472	WATE	024440	W13	064322
TS76	005720	UIPDR2=	177604	UUUTP1	106460	WATE1	024372	W14	064330
TS77	005754	UIPDR3=	177606	UUU10	106470	WATE2	024406	W15	064342
TTBDON	113420	UIPDR4=	177610	UUU2	106406	WATE3	024434	W16	064366
TTBTP1	113376	UIPDR5=	177612	UUU3	106432	WBIT =	000100	W17	064412
TTBTP2	113406	UIPDR6=	177614	U0	063330	WC	051540	W2	064122
TTB10	113416	UIPDR7=	177616	U1	063352	WDAP00	064442	W20	064420
TTB2	113332	UNIQUE	131610	U10	063642	WDAT01	064444	W3	064146
TTCDON	122170	UPAT00	064016	U11	063644	WDAT02	064446	W4	064154
TTCSR =	177564	UPAT01	064020	U12	063662	WDAT03	064450	W5	064166
TTCTB0	122146	UPAT02	064022	U13	063700	WDONE	064452	W6	064212
TTCTB1	122152	UPAT03	064024	U14	063732	WDONE2	130330	W7	064236
TTCTB2	122162	UPAT10	064026	U15	063760	WPAT00	064432	X	124436
TTC10	122166	UPAT11	064030	U16	063762	WPAT01	064434	XAPT11	065030

```
XBUF    104750   XT4     104306   YDAT01  065240   Y3      065144   $DEVCT  001010
XDAT00  065006   XT4A    104336   YDAT02  065242   Y4      065164   $DOAGN  133260
XDAT01  065010   XT4B    104350   YDAT03  065244   Y5      065224   $ENDAD  133212
XDAT02  065012   XT5     104366   YDONE   065276   Z       124442   $ENV    001020
XDAT03  065014   XT5A    104406   YFLAG   065226   ZDAT00  065450   $ENVM   001021
XDONE   065046   XT5B    104420   YNTAB   021300   ZDAT01  065452   $EOPCT  133146
XMIT2   132210   XT6     104436   YPAT00  065246   ZDAT02  065454   $ETABL  001020
XMTCT2  132442   XT6A    104462   YPAT01  065250   ZDAT03  065456   $ETEND  001030
XOR1    016226   XT6B    104474   YPAT02  065252   ZDONE   065510   $FATAL  001002
XOR2    016230   XT7     104512   YPAT03  065254   ZFLAG   065442   $GDADR  131500
XOR3    016256   XT8     104540   YPAT10  065256   ZPAT00  065460   $GDDAT  131502
XPAT0   105050   XT9     104566   YPAT11  065260   ZPAT01  065462   $HIBTS  001030
XPAT0   104760   XXBDON  113716   YPAT12  065262   ZPAT02  065464   $MAIL   001000
XPAT00  065016   XXBTP1  113674   YPAT13  065264   ZPAT03  065466   $MBADR  001032
XPAT01  065020   XXBTP2  113704   YPAT20  065266   ZPAT10  065470   $MSGAD  001014
XPAT02  065022   XXB10   113714   YPAT21  065270   ZPAT11  065472   $MSGLG  001016
XPAT03  065024   XXB2    113622   YPAT22  065272   ZPAT12  065474   $MSGTY  001000
XPAT1   104770   XXCDON  123464   YPAT23  065274   ZPAT13  065476   $PASS   001006
XPAT10  065026   XXXDON  107436   YTMP1   065230   ZPAT20  065500   $PASTM  001036
XPAT12  065032   XXX1    107010   YTMP2   065232   ZPAT21  065502   $SETUP= 000020
XPAT13  065034   XXX2    107054   YTMP3   065234   ZPAT22  065504   $STUP = 177777
XPAT2   105000   XXX3    107120   YYBDON  114070   ZPAT23  065506   $SVPC = 000400
XPAT20  065036   XXX4    107164   YYBTP1  114044   ZTMP1   065444   $SWR  = 000000
XPAT21  065040   XXX5    107230   YYBTP2  114054   ZTMP2   065446   $SWREG  001022
XPAT22  065042   X10     064616   YYBTP3  114064   ZZCBF   124116   $TESTN  001004
XPAT23  065044   X11     064632   YYB10   114066   ZZCDON  124130   $TMP0   037360
XPAT3   105010   X12     064652   YYB2    113772   ZZC10   124112   $TMP1   037362
XPAT4   105020   X13     064676   YYCDON  124022   ZZC2    124034   $TMP2   037364
XPAT5   105030   X14     064704   YYC1    123470   ZZC3    124054   $TMP3   037366
XPAT6   105040   X15     064720   YYC2    123516   ZZZDON  110130   $TMP4   037370
XTDONE  105060   X16     064740   YYC3    123544   ZZZ10   110126   $TN   = 000663
XT1     104166   X17     064764   YYC4    123572   ZZZ2    110102   $TPB    025460
XT1A    104202   X2      064476   YYC5    123620   Z1      065322   $TPS    025462
XT10    104612   X20     064772   YYC6    123646   Z2      065342   $TSTM   001034
XT11    104640   X3      064522   YYYDON  110070   Z3      065366   $UNIT   001012
XT12    104666   X4      064530   YYY1    107442   Z4      065374   $UNITM  001040
XT13    104722   X5      064544   YYY2    107506   Z5      065406   $USWR   001024
XT2     104204   X6      064564   YYY3    107552   Z6      065440   $X    = 131766
XT2A    104224   X7      064610   YYY4    107616   $APTHD  001030   .     = 133453
XT2B    104242   Y       124440   YYY5    107662   $BDADR  131474   .RSET   124572
XT3     104254   YBR     001602   Y1      065100   $BDDAT  131476   .$X   = 001030
XT3A    104276   YDAT00  065236   Y2      065120   $CPUOP  001026

. ABS.  133453    000    CON   RW   REL   LCL   I

ERRORS DETECTED:  0

CJKDJB,CJKDJB/SOL/NL:TOC=SYSMAC.SML,CJKDJB.P11
RUN-TIME: 65 77 3 SECONDS
RUN-TIME RATIO: 273/145=1.8
CORE USED:  48K  (95 PAGES)
```