FP11

DIAGNOSTIC NO. 2

MD-11-DFFPB-A

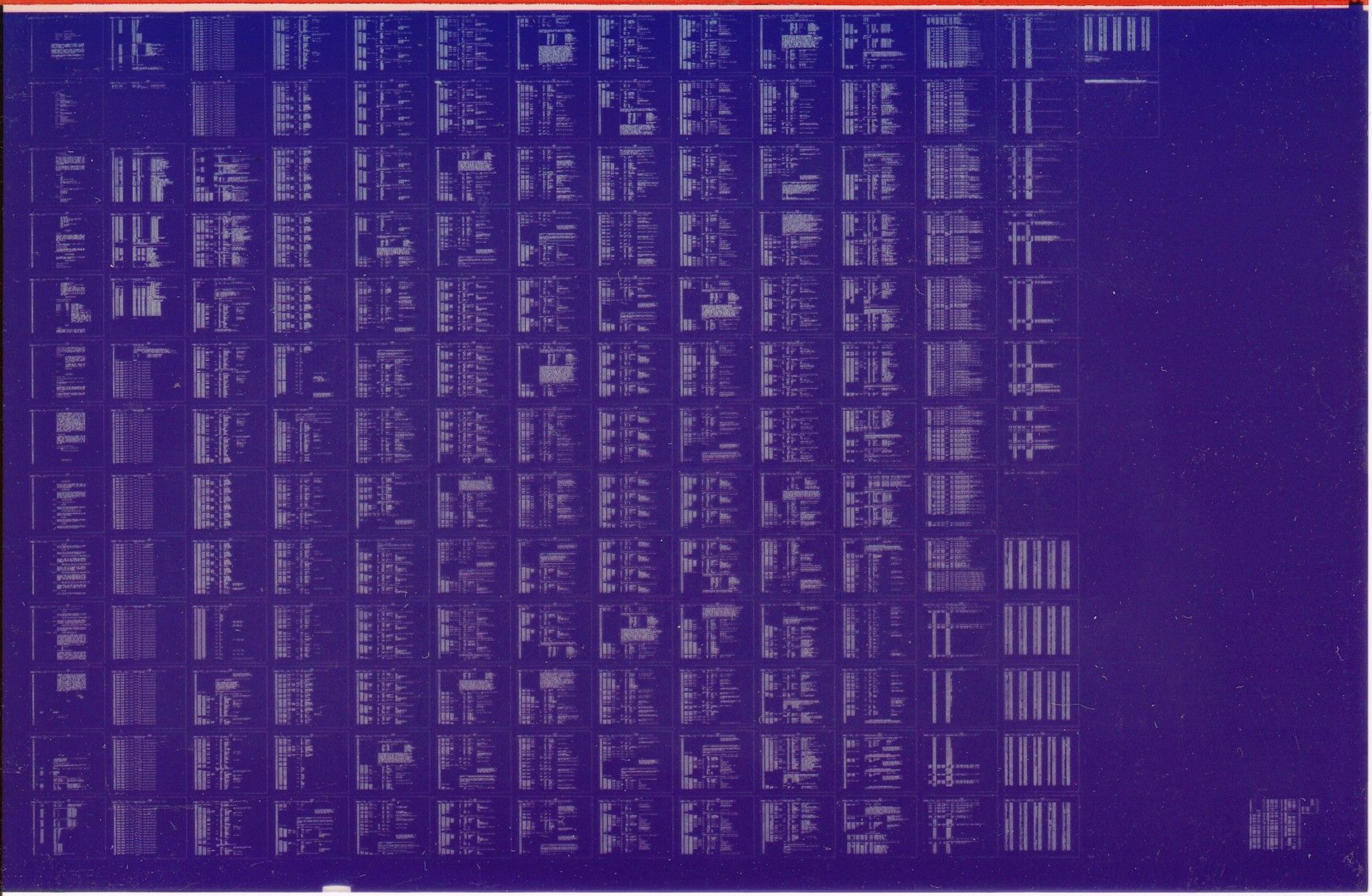
EP-DFFPB-A-DL-A

FICHE 1 OF 1

DEC 1976

digital

MADE IN USA



.REM &

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DFFPB-A-D

PRODUCT NAME: PDP-11/34 FPP DIAGNOSTIC PART 2

DATE: DECEMBER 1976

AUTHOR: ANTHONY VEZZA

CONTENTS

1. ABSTRACT

THE THREE PROGRAMS:

DFFPA DFFPB DFFPC

ARE DESIGN TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/34 FP11-A FLOATING POINT PROCESSOR. THE DESIGN IS AN ATTEMPT TO REACH ALL ROM STATES, TAKE ALL BRANCH MICRO TESTS (BUT'S) AND VARIFY ALL THE LOGIC. THEY CONSIST OF 155 (OCT) INDIVIDUAL TESTS SEQUENCED TO DETECT AND ATTEMPT TO IDENTIFY FAULTS WITH A MINIMUM HARDWARE OR SOFTWARE LEVEL. THE TESTS ARE PARTIONED INTO THREE STAND-ALONE PROGRAMS DESCRIBED BELOW.

NOTE THAT ERROR REPORTS IN THESE PROGRAMS ARE BASED UPON THE KNOWLEDGE THAT ALL PREVIOUS TESTS HAVE BEEN RUN AND IN MOST CASE THAT THERE IS ONLY A SINGLE POINT FAULT IN THE FP11-A. IF THE PROGRAMS OR TESTS ARE NOT RUN IN ORDER THEN ERROR MESSAGES MAY NOT BE ACCURATE.

A. DFFPA

DFFPA TESTS:

- LDFPS
- STFPS
- CFCC
- SETF, SETD, SETI AND SETL
- STST
- LDF AND LDD (ALL SOURCE MODES)
- STD (MODE 0 AND 1)
- ADDF, ADDD AND SUBD (MOST CONDITIONS)

B. DFFPB

DFFPB TESTS:

- ADDF, ADDD AND SUBD (ALL CONDITIONS NOT TESTED IN DFFPA)
- CMPD AND CMPF
- DIVD AND DIVF
- MULD AND MULF
- MODD AND MODF

C. DFFPC

DFFPC TESTS:

- STF AND STD (ALL MODES)
- STCFD AND STCDF
- CLRD AND CLRF
- NEGF AND NEGD

ABSF AND ABSD
TSTF AND TSTD
NEGF, ABSF AND TSTF (ALL SOURCE MODES)
NEGF, ABSF AND TSTF (ALL SOURCE MODES,
LDFPS (ALL SOURCE MODES)
LDCIF AND LDCLF
LDCID AND LDCLD
LDEXP
STFPS (ALL DESTINATION MODES)
STCFL AND STCFI
STCDL AND STCDI
STEXP
STST

2. REQUIREMENTS

2.1 EQUIPMENT

A PDP 11/34 (WITH OR WITHOUT CONSOLE), LA30 (OR EQUIVALENT) AND AN FP11-A FLOATING POINT PROCESSOR. NOTE THAT A SPECIAL INTERRUPTS TEST MODULE IS BEING DESIGNED FOR USE IN THE MANUFACTURING ENVIRONMENT. WHEN THIS DEVICE IS PRESENT THE PROGRAM DFFPB WILL MAKE USE OF IT TO TEST THE FPP INTERRUPT ON BUS REQUEST FUNCTIONS.

2.2 STORAGE

ALL THREE PROGRAM REQUIRE A MEMORY SYSTEM OF AT LEAST 16K TO LOAD AND RUN.

2.3 PRELIMINARY PROGRAMS

THESE THREE DIAGNOSTICS WILL ASSUME THAT THE PDP 11/34 CENTRAL PROCESSOR IS FAULTLESS, THEREFORE WHEN IN DOUBT RUN THE PDP 11/34 PROCESSOR DIAGNOSTICS BEFORE THESE FP11-A DIAGNOSTICS.

3. LOADING PROCEDURE

THE PROGRAMS WILL BE SUPPLIED ON THE 11/34 DIAGNOSTIC MEDIA. REFER TO THE XXDP OPERATING MANUAL FOR FURTHER INFORMATION.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 PROGRAM AND OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY
2. LOAD ADDRESS 200
3. SET CONSOLE SWITCHES (IF CONSOLE IS PRESENT)
4. PRESS START
   ON FIRST PASS THE PROGRAM WILL IDENTIFY ITSELF. NOTE THAT IF THERE IS NO PHYSICAL CONSOLE THE PROGRAM WILL REQUEST THE OPERATOR FOR INITIAL VALUE FOR THE SOFTWARE SWITCH REGISTER (SEE SECTION 8.5). IF RUNNING UNDER ACT, APT OR CHAIN THIS DOES NOT APPLY.
5. THE PROGRAM WILL LOOP AND AN END OF PASS AND ERROR SUMMARY WILL BE TYPED AT THE END OF EVERY PASS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE SWITCH SETTING ARE:

	OCTAL	
SW(15)=1...	100000	HALT ON ERROR
SW(14)=1...	40000	LOOP ON CURRENT TEST
SW(13)=1...	20000	INHIBIT ERROR TYPE OUTS
SW(12)=1...	10000	INHIBIT T-BIT TRAPPING
SW(11)=1...	4000	INHIBIT ITERATIONS
SW(10)=1...	2000	RING TTY BELL ON ERROR
SW(9)=1....	1000	LOOP ON ERROR
SW(8)=1....	400	LOOP ON TEST SPECIFIED IN SW(5) THROUGH SW(0)
SW(7)=1....	200	PRINT ERROR SUMMARY EVEN IF SW(13)=1. THIS APPLIES ONLY TO PROGRAM DFFPA.
SW(7)=1....	200	DESELECT CORRECT INTERRUPT TEST IN PROGRAM DFFPB. NOTE THAT THIS TEST WILL AUTOMATICALLY BE DESELECTED BY THE ABSENCE OF THE SPECIAL TEST EQUIPMENT DESIGNED TO CONDUCT THIS TEST. IF THIS EQUIPMENT IS NOT INSTALLED THERE IS NO NEED TO DESELECT THIS TEST. THIS APPLIES ONLY TO PROGRAM DFFPB!

6. ERRORS

6.1 SUMMARIES

IN PROGRAM DFFPA TESTS 1 AND 11 HAVE A SPECIAL ERROR SUMMARY FEATURE. THESE TWO TEST RUN MANY TEST PATTERNS THROUGH THE LOGIC. AFTER AN ERROR IS ENCOUNTERED, ONLY THE FIRST FIVE ERRORS ARE REPORTED

(TYPED ON THE TTY). EVERY ERROR THOUGH IS LOGGED AND AN ERROR SUMMARY IS PRINTED WHEN THE TEST IS COMPLETE. NOTE THAT IS SW<13>=1 THIS SUMMARY WILL NOT BE TYPED UNLESS SW<7>=1. IN OTHER WORDS TO GET JUST AN ERROR SUMMARY FROM EITHER OF THESE TWO TESTS 1 AND 11 IN PROGRAM DFFPA BOTH SWITCHES 13 AND 7 MUST = 1.

6.2 ERROR RECOVERY

SW<15:9>=0... MOST ERRORS WILL CAUSE EXECUTION TO GO TO THE START OF THE NEXT TEST AFTER THE MESSAGE IS TYPED. A FEW TESTS ARE IN SECTIONS. IN THESE TESTS AN ERROR WILL CAUSE EXECUTION TO GO TO THE NEXT SECTION AFTER THE MESSAGE IS TYPED.

SW<15>=1... THE PROGRAM WILL HALT AFTER TYPING THE ERROR MESSAGE. PRESSING THE CONSOLE CONTINUE WILL CAUSE THE PROGRAM TO CONTINUE AS IF SW<15>=0.

7. RESTRICTIONS

NONE

8. MISCELLANEOUS

8.1 EXECUTION TIMES

LESS THAN 10 SECONDS FOR EACH PROGRAM ON ANY PASS.

8.2 STACK POINTER

THE STACK POINTER IS INITIALIZED TO 1100 IN EACH OF THE THREE PROGRAMS.

8.3 PASS COUNT

THE PROGRAM MAKES ONE PASS FOR EACH END OF PASS MESSAGE TYPED. THE END OF PASS MESSAGE DESCRIBES THE TOTAL NUMBER OF PASSES COMPLETED AND THE TOTAL NUMBER OF ERRORS SINCE THE LAST END OF PASS MESSAGE.

8.4 T-BIT TRAPPING

IF SW<12>=0 EACH PROGRAM WILL RUN WITH TRACE TRAPS ON EVERY OTHER PASS. FIRST PASS WILL NOT ENABLE TRACE TRAPS. NOTE SW<12>=1 DISABLES T-BIT TRAPS.

8.5 SOFTWARE SWITCH REGISTER

EACH OF THE THREE PROGRAMS WILL RUN WITH OR WITHOUT A CONSOLE SWITCH REGISTER. IF A PHYSICAL CONSOLE SWITCH REGISTER IS PRESENT ON THE SYSTEM, THEN THESE PROGRAMS WILL GO AHEAD AND USE IT FOR THE SWITCH FUNCTIONS DESCRIBED IN 5.1 ABOVE. IF HOWEVER THERE IS NO CONSOLE SWITCH REGISTER ON THE SYSTEM A SOFTWARE SWITCH REGISTER WILL BE USED. THIS SOFTWARE SWITCH REGISTER CAN BE EXAMINED OR MODIFIED AT ANY TIME BY THE USER IF HE TYPES CONTROL G WHILE THE PROGRAM IS RUNNING. THIS CONTROL G WILL CAUSE THE CONTENTS OF THE SOFTWARE SWITCH REGISTER TO BE TYPED ON THE TTY AND ASK THE USER FOR A NEW VALUE. WHEN THE USER TYPES A VALUE AND CARRIAGE RETURN THEN THE PROGRAM WILL RESUME TESTING AT THE SAME POINT AT WHICH IT LEFT OFF WHEN THE USER TYPED CONTROL G. NOTE THAT WHEN NOT RUNNING UNDER ACT, APT OR CHAIN THE USER WILL BE ASKED FOR A SOFTWARE SWITCH REGISTER VALUE AFTER LOADING ADDRESS 200 AND STARTING THE PROGRAM THE FIRST TIME THE PROGRAM IS RUN AFTER LOADING (ONLY IF NO CONSOLE SWITCH REGISTER IS ON THE SYSTEM).

8.6 INTERRUPTS TEST

IN PROGRAM DFFPB THERE IS A SPECIAL TEST FOR CHECKING THE CORRECT FLOWS OF THE FPP. THIS TEST CAN BE RUN ONLY IF A SPECIAL TEST MODULE IS IN THE SYSTEM. THIS MODULE WILL PROBABLY ONLY BE USED IN MANUFACTURING. IF THIS MODULE IS NOT IN THE SYSTEM THIS TEST WILL AUTOMATICALLY BE DESELECTED. IF THIS TEST MODULE IS ON THE SYSTEM AND SW<7>=0 THIS TEST WILL BE RUN. IF SW<7>=1 THIS TEST WILL BE DESELECTED.

8.7 ACT, APT AND XXDP COMPATIBILITY

THESE PROGRAMS ARE FULLY COMPATIBLE WITH:

- APT
- ACT
- XXDP MONITOR AND CHAIN PROGRAMS.

9. PROGRAM DESCRIPTION

TEST 1 ROUND\TRUNK TEST

THIS IS A TEST OF THE ROUND\TRUNK FLOWS. IN PARTICULAR TWO THINGS ARE TESTED: FIRST A CONDITION IN WHICH ROUNDING RESULTS IN THE NEED FOR RENORMALIZATION, AND SECOND THE PSW CONDITION CODES N AND Z BIT COMBINATIONS

TEST 2 OVER\UNDER TEST

THIS IS A PARTIAL TEST OF THE OVER\UNDER FLOWS. ONE OVERFLOW AND TWO UNDERFLOW CONDITIONS ARE CHECKED. THE REMAINING UNDERFLOW COND. AND THE REMAINING OVERFLOW COND. WILL BE CHECKED LATER USING THE XXX INSTRUCTION. HERE EACH CONDITION TESTED IS CHECKED BOTH WITH TRAPS ENABLED (FIU=1 OR FIV=1) AND ALSO WITH TRAPS DISABLED (FIU=0 OR FIV=0).

TEST 3 LDCFD AND LDCDF TEST

THIS IS A TEST OF LDCFD AND LDCDF.

TEST 4 CMPD TEST

THIS IS A TEST OF THE CMPD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS

TEST 5 DIVD WITH (FSRC=0) AND (BUT FD) TEST

THIS IS A TEST OF THE DIVD INSTRUCTION WITH A ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH TRAP ENABLED AND TRAPS DISABLED.

TEST 6 DIVF TEST

THIS IS A TEST OF THE DIVF INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 7 DIVD TEST

THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 10 MULF TEST

THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS.

TEST 11 MULD TEST

THIS IS A TEST OF THE MULD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 12 UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS.

TEST 13 UNDER\OVER FLOW, USING MULD WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN ARRISE USING THE MULD INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 14 UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST

THIS IS A TEST OF THE UNDERFLOW AND OVERFLOW CONDITIONS THAT CAN OCCUR USING THE MULF INSTRUCTION. A SUBROUTINE IS CALLED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS. HERE THE PARTICULAR INTERRUPT, EITHER OVERFLOW OR UNDERFLOW, IS ENABLED SO A TRAP SHOULD OCCUR.

TEST 15 UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST

THIS IS A TEST OF THE OVER FLOW AND UNDER FLOW CONDITIONS USING THE MULD INSTRUCTION WITH TRAPS ENABLED. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 16 MODF TEST

THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE

THE MODF INSTRUCTION AND CHECK THE RESULTS.

TEST 17 MODD TEST

THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 20 UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.

TEST 21 UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE MODD INSTRUCTION'S OVER FLOW AND UNDER FLOW CONDITIONS. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MODD INSTUCTION AND CHECK THE RESULTS.

TEST 22 INTERRUPT CORRECT FLOWS TEST

THIS IS A TEST OF THE 'CORRECT' FLOWS. THIS PART OF THE MICRO CODE HAS AS ITS PURPOSE INSURING THAT INTERRUPT REQUESTS MADE DURING CERTAIN LENGTHY FPP INSTRUCTIONS GET HONORED. THIS IS DONE IN A WAY SUCH THAT IF AN INTERRUPT REQUEST OCCURS DURING ONE OF THESE INSTRUCTIONS THE STATE OF THAT INSTRUCTION'S EXECUTION WILL BE THE SAME AS IF THAT INSTRUCTION HAD NEVER BEEN FETCHED AND ITS EXECUTION NEVER STARTED. THUS THE MICRO CODE WILL RESTORE ALL REGISTERS, BACK UP THE PC AND LEAVE THE FPS AND AC0 THROUGH AC5 UNMODIFIED.
THE INSTRUCTIONS FOR WHICH THIS IS NECESSARY ARE:

ADD (OR SUB)
DIV
MUL
MOD
(BOTH DOUBLE AND FLOATING)

ALL ADDRESSING MODES WILL BE TRIED WITH THE ADDD INSTRUCTION. THEN EACH OF THE OTHER INSTRUCTIONS WILL BE TRIED USING MODE 1. NOTE THAT THIS TEST NEEDS A SPECIAL INTERRUPT MODULE, WHICH WILL PROBABLY ONLY BE PRESENT IN DEC'S MANUFACTURING ENVIRONMENT, TO RUN. THIS SPECIAL EQUIPMENT IS DESIGNED TO RAISE AN INTERRUPT REQUEST IN THE PROCESSOR IF A BIT IS SET IN ITS STATUS REGISTER AND ONLY WHEN AN FPP INSTRUCTION IS ENCOUNTERED.

```
                                THEREFORE  THIS  TEST  WILL  BE  RUN  CONDITIONALLY
                                (DEPENDENT UPON WHETHER OR NOT THE  STATUS  REGISTER
                                OF  THE  TEST  EQUIPMENT TIMES OUT WHEN REFERENCED).
                                THIS TEST CAN ALSO BE DESELECTED BY TURNING SWITCH 7
                                OF  THE  SWITCH  REGISTER  (PHYSICAL OR VIRTUAL) ON.
                                THE TEST ASSUMES THAT THE  TEST  EQUIPMENT'S  STATUS
                                REGISTER  IS  AT  LOCATION  777774  (NOTE  THAT  ALL
                                REFERENCES  TO  THIS  LOCATION  ARE  MADE   INDIRECT
                                THROUGH  THIS  PROGRAMS  LOCATION CORINT, SO THAT IF
                                THE USER HAS MODIFIED THE  TEST  EQUIPMENT'S  STATUS
                                REGISTER  TO RESPOND TO A DIFFERENT ADDRESS LOCATION
                                CORINT  MUST  BE  MADE  TO   CONTAIN   THAT   STATUS
                                REGISTER'S  NEW ADDRESS).  THIS PROGRAM ASSUMES THAT
                                THE TRAP VECTOR  FOR  THE  TEST  EQUIPMENT  IS  110.
                                AGAIN  NOTE  THAT ALL REFERENCES TO THIS TRAP VECTOR
                                ARE INDIRECT, THROUGH THIS PROGRAM'S LOCATION CORTRP
                                (IF  THE  TEST  EQUIPMENT  IS  MADE  TO  TRAP  TO  A
                                DIFFERENT VECTOR LOCATION CORTRP  MUST  CONTAIN  THE
                                ADDRESS OF THIS VECTOR).



                                10.                     LISTING
                                                        -------



                                8

000266                          MNUMBER=266
000002                          PROGNUM=2


                                        .LIST   ME
                                        .NLIST  MD,MC,CND
```

```
                                        .ENABL  ABS


                                .TITLE  MAINDEC-11-DFFPB-A      PDP 11/34 FPP DIAGNOSTIC PART 2
                                ;*COPYRIGHT (C) SEP 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY ANTHONY S. VEZZA
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT


000244                          FPVECT=244
177400                          $SWR=177400
000200                          $SWRMSK=200
000011                          TAB=11
000015                          CRLF=15

                                .SBTTL  BASIC DEFINITIONS

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

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
```

```
                177772          PIRQ=   177772          ;;PROGRAM INTERRJPT REQUEST REGISTER
                177570          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
                177570          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

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

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000          SW15=   100000
040000          SW14=   40000
020000          SW13=   20000
010000          SW12=   10000
004000          SW11=   4000
002000          SW10=   2000
001000          SW09=   1000
000400          SW08=   400
000200          SW07=   200
000100          SW06=   100
000040          SW05=   40
000020          SW04=   20
000010          SW03=   10
000004          SW02=   4
000002          SW01=   2
000001          SW00=   1
                .EQUIV  SW09,SW9
                .EQUIV  SW08,SW8
                .EQUIV  SW07,SW7
                .EQUIV  SW06,SW6
                .EQUIV  SW05,SW5
                .EQUIV  SW04,SW4
                .EQUIV  SW03,SW3
                .EQUIV  SW02,SW2
                .EQUIV  SW01,SW1
                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000          BIT15=  100000
```

```
                        040000          BIT14=  40000
                        020000          BIT13=  20000
                        010000          BIT12=  10000
                        004000          BIT11=  4000
                        002000          BIT10=  2000
                        001000          BIT09=  1000
                        000400          BIT08=  400
                        000200          BIT07=  200
                        000100          BIT06=  100
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          BIT00=  1
                                        .EQUIV  BIT09,BIT9
                                        .EQUIV  BIT08,BIT8
                                        .EQUIV  BIT07,BIT7
                                        .EQUIV  BIT06,BIT6
                                        .EQUIV  BIT05,BIT5
                                        .EQUIV  BIT04,BIT4
                                        .EQUIV  BIT03,BIT3
                                        .EQUIV  BIT02,BIT2
                                        .EQUIV  BIT01,BIT1
                                        .EQUIV  BIT00,BIT0

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

                                        .SBTTL  TRAP CATCHER

                                        .=0
                        000000          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                  .=174
```

```
                        DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
                        SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                        .SBTTL  STARTING ADDRESS(ES)
000336                          JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000          $TSTNM:    .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000          $ERFLG:    .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000       $ICNT:     .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000       $LPADR:    .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000       $LPERR:    .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000       $ERTTL:    .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000          $ITEMB:    .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001          $ERMAX:    .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000       $ERRPC:    .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000       $GDADR:    .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000       $BDADR:    .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000       $GDDAT:    .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000       $BDDAT:    .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000          $AUTOB:    .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000          $INTAG:    .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570       $SWR:      .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570       DISPLAY:   .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560       $TKS:      177560                  ;;TTY KBD STATUS
001146  177562       $TKB:      177562                  ;;TTY KBD BUFFER
001150  177564       $TPS:      177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566       $TPB:      177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000          $NULL:     .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002          $FILLS:    .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012          $FILLC:    .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000          $TPFLG:    .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000       $REGAD:    .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000       $REG0:     .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000       $REG1:     .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000       $REG2:     .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000       $REG3:     .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000       $REG4:     .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000       $REG5:     .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000       $REG6:     .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000       $REG7:     .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000       $REG10:    .WORD   0               ;;CONTAINS (($REGAD)+20)
001204  000000       $REG11:    .WORD   0               ;;CONTAINS (($REGAD)+22)
001206  000000       $REG12:    .WORD   0               ;;CONTAINS (($REGAD)+24)
001210  000000       $REG13:    .WORD   0               ;;CONTAINS (($REGAD)+26)
001212  000000       $REG14:    .WORD   0               ;;CONTAINS (($REGAD)+30)
001214  000000       $REG15:    .WORD   0               ;;CONTAINS (($REGAD)+32)
001216  000000       $REG16:    .WORD   0               ;;CONTAINS (($REGAD)+34)
001220  000000       $REG17:    .WORD   0               ;;CONTAINS (($REGAD)+36)
001222  000000       $REG20:    .WORD   0               ;;CONTAINS (($REGAD)+40)
```

```
945  001224  000000                $REG21: .WORD   0               ;;CONTAINS ((SREGAC)+42)
946  001226  000000                $REG22: .WORD   0               ;;CONTAINS ((SREGAD)+44)
947  001230  000000                $REG23: .WORD   0               ;;CONTAINS ((SREGAD)+46)
948  001232  000000                $TMP0:  .WORD   0               ;;USER DEFINED
949  001234  000000                $TMP1:  .WORD   0               ;;USER DEFINED
950  001236  000000                $TMP2:  .WORD   0               ;;USER DEFINED
951  001240  000000                $TMP3:  .WORD   0               ;;USER DEFINED
952  001242  000000                $TMP4:  .WORD   0               ;;USER DEFINED
953  001244  000000                $TMP5:  .WORD   0               ;;USER DEFINED
954  001246  000000                $TMP6:  .WORD   0               ;;USER DEFINED
955  001250  000000                $TMP7:  .WORD   0               ;;USER DEFINED
956  001252  000000                $TMP10: .WORD   0               ;;USER DEFINED
957  001254  000000                $TMP11: .WORD   0               ;;USER DEFINED
958  001256  000000                $TMP12: .WORD   0               ;;USER DEFINED
959  001260  000000                $TMP13: .WORD   0               ;;USER DEFINED
960  001262  000000                $TMP14: .WORD   0               ;;USER DEFINED
961  001264  000000                $TMP15: .WORD   0               ;;USER DEFINED
962  001266  000000                $TMP16: .WORD   0               ;;USER DEFINED
963  001270  000000                $TMP17: .WORD   0               ;;USER DEFINED
964  001272  000000                $TMP20: .WORD   0               ;;USER DEFINED
965  001274  000000                $TMP21: .WORD   0               ;;USER DEFINED
966  001276  000000                $TMP22: .WORD   0               ;;USER DEFINED
967  001300  000000                $TMP23: .WORD   0               ;;USER DEFINED
968  001302  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
969  001304  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
970  001306  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
971  001312     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
972  001313     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
973  001314  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
974                                ;;**************************************************************
975                                .SBTTL  APT MAILBOX-ETABLE
976
977                                ;;**************************************************************
978                                .EVEN
979  001316                        $MAIL:                          ;;APT MAILBOX
980  001316  000000                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
981  001320  000000                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
982  001322  000000                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
983  001324  000000                $PASS:  .WORD   APASS           ;;PASS COUNT
984  001326  000000                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
985  001330  000000                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
986  001332  000000                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
987  001334  000000                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
988  001336                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
989  001336     000                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
990  001337     000                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
991  001340  000000                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
992  001342  000000                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
993  001344  000000                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
994                                ;*                              BITS 15-11=CPU TYPE
995                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11 45=05
996                                ;*                                      11/70=06,PDQ=07,Q=10
997                                ;*                              BIT 10=REAL TIME CLOCK
998                                ;*                              BIT  9=FLOATING POINT PROCESSOR
999                                ;*                              BIT  8=MEMORY MANAGEMENT
1000 001346     000                $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
```

```
                        $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                                MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              500 NSEC MOS=003
                        ;*
                        $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                        ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
                        $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
                        $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
                        $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
                        $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
                        $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
                        $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
                        $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
                        $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
                        $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
                        $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
                        $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
                        $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
                        $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
                        $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
                        $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
                        $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
                        $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
                        $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
                        $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
                        $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
                        $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
                        $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
                        $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
                        $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
                        $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
                        $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
                        $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
                        $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
                        $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
                        $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
                        $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15


                        $ETEND:
```

```
                                        .SBTTL  ERROR POINTER TABLE

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

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

001442                                  $ERRTB:
                                        ;ITEM 1
001442  037660  066216  070540                  .WORD   EM1,DH1,DT1,DF1
001450  070012
                                        ;ITEM 2
001452  037712  066306  070576                  .WORD   EM2,DH2,DT2,DF2
001460  070030
                                        ;ITEM 3
001462  037746  066216  070540                  .WORD   EM3,DH3,DT3,DF3
001470  070012
                                        ;ITEM 4
001472  040053  066216  070540                  .WORD   EM4,DH4,DT4,DF4
001500  070012
                                        ;ITEM 5
001502  040160  066216  070540                  .WORD   EM5,DH5,DT5,DF5
001510  070012
                                        ;ITEM 6
001512  040265  066216  070540                  .WORD   EM6,DH6,DT6,DF6
001520  070012
                                        ;ITEM 7
001522  040372  066216  070540                  .WORD   EM7,DH7,DT7,DF7
001530  070012
                                        ;ITEM 10
001532  040477  066216  070540                  .WORD   EM10,DH10,DT10,DF10
001540  070012
                                        ;ITEM 11
001542  040604  066216  070540                  .WORD   EM11,DH11,DT11,DF11
001550  070012
                                        ;ITEM 12
001552  040713  066216  070540                  .WORD   EM12,DH12,DT12,DF12
001560  070012
                                        ;ITEM 13
001562  041022  066216  070540                  .WORD   EM13,DH13,DT13,DF13
001570  070012
                                        ;ITEM 14
001572  041136  066216  070540                  .WORD   EM14,DH14,DT14,DF14
001600  070012
                                        ;ITEM 15
001602  041250  066216  070540                  .WORD   EM15,DH15,DT15,DF15
001610  070012
                                        ;ITEM 16
001612  041362  066216  070540                  .WORD   EM16,DH16,DT16,DF16
```

```
  999  001620  070012
 1000                                   ;ITEM 17
 1001  001622  041457  066347  070624          .WORD   EM17,DH17,DT17,DF17
 1002  001630  070042
 1003                                   ;ITEM 20
 1004  001632  041534  066216  070644          .WORD   EM20,DH20,DT20,DF20
 1005  001640  070051
 1006                                   ;ITEM 21
 1007  001642  041566  066422  070666          .WORD   EM21,DH21,DT21,DF21
 1008  001650  070051
 1009                                   ;ITEM 22
 1010  001652  041620  066511  070710          .WORD   EM22,DH22,DT22,DF22
 1011  001660  070051
 1012                                   ;ITEM 23
 1013  001662  041701  066216  070726          .WORD   EM23,DH23,DT23,DF23
 1014  001670  070061
 1015                                   ;ITEM 24
 1016  001672  041756  066216  070726          .WORD   EM24,DH24,DT24,DF24
 1017  001700  070061
 1018                                   ;ITEM 25
 1019  001702  042054  066216  070726          .WORD   EM25,DH25,DT25,DF25
 1020  001710  070061
 1021                                   ;ITEM 26
 1022  001712  042141  066216  070726          .WORD   EM26,DH26,DT26,DF26
 1023  001720  070061
 1024                                   ;ITEM 27
 1025  001722  042054  066216  070726          .WORD   EM27,DH27,DT27,DF27
 1026  001730  070061
 1027                                   ;ITEM 30
 1028  001732  042237  066216  070726          .WORD   EM30,DH30,DT30,DF30
 1029  001740  070061
 1030                                   ;ITEM 31
 1031  001742  042311  066216  070726          .WORD   EM31,DH31,DT31,DF31
 1032  001750  070061
 1033                                   ;ITEM 32
 1034  001752  041724  066216  070726          .WORD   EM32,DH32,DT32,DF32
 1035  001760  070061
 1036                                   ;ITEM 33
 1037  001762  042356  066216  070726          .WORD   EM33,DH33,DT33,DF33
 1038  001770  070105
 1039                                   ;ITEM 34
 1040  001772  042401  066216  070726          .WORD   EM34,DH34,DT34,DF34
 1041  002000  070105
 1042                                   ;ITEM 35
 1043  002002  042433  066216  070726          .WORD   EM35,DH35,DT35,DF35
 1044  002010  070105
 1045                                   ;ITEM 36
 1046  002012  042506  066216  070726          .WORD   EM36,DH36,DT36,DF36
 1047  002020  070105
 1048                                   ;ITEM 37
 1049  002022  042554  066216  070726          .WORD   EM37,DH37,DT37,DF37
 1050  002030  070061
 1051                                   ;ITEM 40
 1052  002032  042577  066216  070726          .WORD   EM40,DH40,DT40,DF40
 1053  002040  070061
 1054                                   ;ITEM 41
```

```
1055  002042  042631  066216  070726          .WORD   EM41,DH41,DT41,DF41
1056  002050  070061
1057                                  ;ITEM 42
1058  002052  042704  066216  070726          .WORD   EM42,DH42,DT42,DF42
1059  002060  070061
1060                                  ;ITEM 43
1061  002062  043035  066216  070726          .WORD   EM43,DH43,DT43,DF43
1062  002070  070061
1063                                  ;ITEM 44
1064  002072  043166  066216  070726          .WORD   EM44,DH44,DT44,DF44
1065  002100  070061
1066                                  ;ITEM 45
1067  002102  043234  066216  070726          .WORD   EM45,DH45,DT45,DF45
1068  002110  070061
1069                                  ;ITEM 46
1070  002112  043307  066216  070726          .WORD   EM46,DH46,DT46,DF46
1071  002120  070105
1072                                  ;ITEM 47
1073  002122  043364  066216  070726          .WORD   EM47,DH47,DT47,DF47
1074  002130  070105
1075                                  ;ITEM 50
1076  002132  043520  066216  070726          .WORD   EM50,DH50,DT50,DF50
1077  002140  070105
1078                                  ;ITEM 51
1079  002142  043573  066216  070726          .WORD   EM51,DH51,DT51,DF51
1080  002150  070105
1081                                  ;ITEM 52
1082  002152  043641  066216  070726          .WORD   EM52,DH52,DT52,DF52
1083  002160  070105
1084                                  ;ITEM 53
1085  002162  051471  066216  071054          .WORD   EM53,DH53,DT53,DF53
1086  002170  070201
1087                                  ;ITEM 54
1088  002172  051522  066216  071054          .WORD   EM54,DH54,DT54,DF54
1089  002200  070201
1090                                  ;ITEM 55
1091  002202  051437  066216  071054          .WORD   EM55,DH55,DT55,DF55
1092  002210  070201
1093                                  ;ITEM 56
1094  002212  051552  066216  071054          .WORD   EM56,DH56,DT56.DF56
1095  002220  070201
1096                                  ;ITEM 57
1097  002222  051643  066216  071054          .WORD   EM57,DH57,DT57,DF57
1098  002230  070201
1099                                  ;ITEM 60
1100  002232  051733  066216  071054          .WORD   EM60,DH60,DT60.DF60
1101  002240  070201
1102                                  ;ITEM 61
1103  002242  052035  066216  071054          .WORD   EM61,DH61,DT61.DF61
1104  002250  070201
1105                                  ;ITEM 62
1106  002252  052231  066216  071054          .WORD   EM62,DH62,DT62,DF62
1107  002260  070201
1108                                  ;ITEM 63
1109  002262  052425  066216  071054          .WORD   EM63,DH63.DT63,DF63
1110  002270  070201
```

```
1111                                            ;ITEM 64
1112  002272  052536  066216  071054                    .WORD   EM64,DH64,DT64,DF64
1113  002300  070201
1114                                            ;ITEM 65
1115  002302  052655  066216  071054                    .WORD   EM65,DH65,DT65,DF65
1116  002310  070201
1117                                            ;ITEM 66
1118  002312  052770  066216  071054                    .WORD   EM66,DH66,DT66,DF66
1119  002320  070201
1120                                            ;ITEM 67
1121  002322  053037  066216  071054                    .WORD   EM67,DH67,DT67,DF67
1122  002330  070201
1123                                            ;ITEM 70
1124  002332  053122  066216  071054                    .WORD   EM70,DH70,DT70,DF70
1125  002340  070233
1126                                            ;ITEM 71
1127  002342  053153  066216  071054                    .WORD   EM71,DH71,DT71,DF71
1128  002350  070233
1129                                            ;ITEM 72
1130  002352  053203  066216  071054                    .WORD   EM72,DH72,DT72,DF72
1131  002360  070233
1132                                            ;ITEM 73
1133  002362  053203  066216  071054                    .WORD   EM73,DH73,DT73,DF73
1134  002370  070233
1135                                            ;ITEM 74
1136  002372  053235  066216  071054                    .WORD   EM74,DH74,DT74,DF74
1137  002400  070233
1138                                            ;ITEM 75
1139  002402  053344  066216  071054                    .WORD   EM75,DH75,DT75,DF75
1140  002410  070233
1141                                            ;ITEM 76
1142  002412  053435  066216  071054                    .WORD   EM76,DH76,DT76,DF76
1143  002420  070233
1144                                            ;ITEM 77
1145  002422  053525  066216  071054                    .WORD   EM77,DH77,DT77,DF77
1146  002430  070233
1147                                            ;ITEM 100
1148  002432  053635  066216  071054                    .WORD   EM100,DH100,DT100,DF100
1149  002440  070233
1150                                            ;ITEM 101
1151  002442  053720  066216  071054                    .WORD   EM101,DH101,DT101,DF101
1152  002450  070233
1153                                            ;ITEM 102
1154  002452  054114  066216  071054                    .WORD   EM102,DH102,DT102,DF102
1155  002460  070233
1156                                            ;ITEM 103
1157  002462  054310  066216  071054                    .WORD   EM103,DH103,DT103,DF103
1158  002470  070233
1159                                            ;ITEM 104
1160  002472  054422  066216  071054                    .WORD   EM104,DH104,DT104,DF104
1161  002500  070233
1162                                            ;ITEM 105
1163  002502  054567  066216  071054                    .WORD   EM105,DH105,DT105,DF105
1164  002510  070233
1165                                            ;ITEM 106
1166  002512  054676  066216  071054                    .WORD   EM106,DH106,DT106,DF106
```

```
1167  002520  070233
1168                                  ;ITEM 107
1169  002522  055005  066216  071054          .WORD   EM107,DH107,DT107,DF107
1170  002530  070233
1171                                  ;ITEM 110
1172  002532  055114  066216  071054          .WORD   EM110,DH110,DT110,DF110
1173  002540  070233
1174                                  ;ITEM 111
1175  002542  043731  066216  070726          .WORD   EM111,DH111,DT111,DF111
1176  002550  070061
1177                                  ;ITEM 112
1178  002552  044006  066216  070726          .WORD   EM112,DH112,DT112,DF112
1179  002560  070061
1180                                  ;ITEM 113
1181  002562  044064  066216  070726          .WORD   EM113,DH113,DT113,DF113
1182  002570  070061
1183                                  ;ITEM 114
1184  002572  044142  066216  070726          .WORD   EM114,DH114,DT114,DF114
1185  002600  070061
1186                                  ;ITEM 115
1187  002602  044221  066557  071000          .WORD   EM115,DH115,DT115,DF115
1188  002610  070131
1189                                  ;ITEM 116
1190  002612  044277  066557  071000          .WORD   EM116,DH116,DT116,DF116
1191  002620  070131
1192                                  ;ITEM 117
1193  002622  044356  066557  071000          .WORD   EM117,DH117,DT117,DF117
1194  002630  070131
1195                                  ;ITEM 120
1196  002632  044514  066557  071000          .WORD   EM120,DH120,DT120,DF120
1197  002640  070131
1198                                  ;ITEM 121
1199  002642  044652  066557  071000          .WORD   EM121,DH121,DT121,DF121
1200  002650  070131
1201                                  ;ITEM 122
1202  002652  045007  066557  071000          .WORD   EM122,DH122,DT122,DF122
1203  002660  070131
1204                                  ;ITEM 123
1205  002662  045144  066216  070726          .WORD   EM123,DH123,DT123,DF123
1206  002670  070105
1207                                  ;ITEM 124
1208  002672  045222  066216  070726          .WORD   EM124,DH124,DT124,DF124
1209  002700  070105
1210                                  ;ITEM 125
1211  002702  045301  066216  070726          .WORD   EM125,DH125,DT125,DF125
1212  002710  070105
1213                                  ;ITEM 126
1214  002712  045357  066216  070726          .WORD   EM126,DH126,DT126.DF126
1215  002720  070105
1216                                  ;ITEM 127
1217  002722  045436  066557  071000          .WORD   EM127,DH127,DT127,DF127
1218  002730  070155
1219                                  ;ITEM 130
1220  002732  045514  066557  071000          .WORD   EM130,DH130,DT130,DF130
1221  002740  070155
1222                                  ;ITEM 131
```

```
1223  002742  045573  066557  071000          .WORD   EM131,DH131,DT131,DF131
1224  002750  070155
1225                                  ;ITEM 132
1226  002752  045731  066216  070726          .WORD   EM132,DH132,DT132,DF132
1227  002760  070105
1228                                  ;ITEM 133
1229  002762  046067  066557  071000          .WORD   EM133,DH133,DT133,DF133
1230  002770  070155
1231                                  ;ITEM 134
1232  002772  046225  066557  071000          .WORD   EM134,DH134,DT134,DF134
1233  003000  070155
1234                                  ;ITEM 135
1235  003002  046362  066216  070726          .WORD   EM135,DH135,DT135,DF135
1236  003010  070105
1237                                  ;ITEM 136
1238  003012  046517  066557  071000          .WORD   EM136,DH136,DT136,DF136
1239  003020  070155
1240                                  ;ITEM 137
1241  003022  046654  066557  071000          .WORD   EM137,DH137,DT137,DF137
1242  003030  070131
1243                                  ;ITEM 140
1244  003032  046742  066557  071000          .WORD   EM140,DH140,DT140,DF140
1245  003040  070131
1246                                  ;ITEM 141
1247  003042  043731  066557  071000          .WORD   EM141,DH141,DT141,DF141
1248  003050  070131
1249                                  ;ITEM 142
1250  003052  044006  066557  071000          .WORD   EM142,DH142,DT142,DF142
1251  003060  070131
1252                                  ;ITEM 143
1253  003062  047031  066557  071000          .WORD   EM143,DH143,DT143,DF143
1254  003070  070131
1255                                  ;ITEM 144
1256  003072  047107  066557  071000          .WORD   EM144,DH144,DT144,DF144
1257  003100  070131
1258                                  ;ITEM 145
1259  003102  047166  066216  070726          .WORD   EM145,DH145,DT145,DF145
1260  003110  070061
1261                                  ;ITEM 146
1262  003112  047333  066216  070726          .WORD   EM146,DH146,DT146,DF146
1263  003120  070061
1264                                  ;ITEM 147
1265  003122  047500  066216  070726          .WORD   EM147,DH147,DT147,DF147
1266  003130  070061
1267                                  ;ITEM 150
1268  003132  047644  066216  070726          .WORD   EM150,DH150,DT150,DF150
1269  003140  070061
1270                                  ;ITEM 151
1271  003142  050010  066557  071000          .WORD   EM151,DH151,DT151.DF151
1272  003150  070155
1273                                  ;ITEM 152
1274  003152  050076  066557  071000          .WORD   EM152,DH152,DT152,DF152
1275  003160  070155
1276                                  ;ITEM 153
1277  003162  045144  066557  071000          .WORD   EM153,DH153,DT153.DF153
1278  003170  070155
```

```
1279                                            ;ITEM 154
1280  003172  045222  066557  071000                .WORD   EM154,DH154,DT154,DF154
1281  003200  070155
1282                                            ;ITEM 155
1283  003202  050165  066557  071000                .WORD   EM155,DH155,DT155,DF155
1284  003210  070155
1285                                            ;ITEM 156
1286  003212  050243  066557  071000                .WORD   EM156,DH156,DT156,DF156
1287  003220  070155
1288                                            ;ITEM 157
1289  003222  050322  066216  070726                .WORD   EM157,DH157,DT157,DF157
1290  003230  070105
1291                                            ;ITEM 160
1292  003232  050467  066557  071000                .WORD   EM160,DH160,DT160,DF160
1293  003240  070155
1294                                            ;ITEM 161
1295  003242  050625  066216  070726                .WORD   EM161,DH161,DT161,DF161
1296  003250  070105
1297                                            ;ITEM 162
1298  003252  050772  066216  070726                .WORD   EM162,DH162,DT162,DF162
1299  003260  070105
1300                                            ;ITEM 163
1301  003262  051136  066557  071000                .WORD   EM163,DH163,DT163,DF163
1302  003270  070155
1303                                            ;ITEM 164
1304  003272  051273  066216  070726                .WORD   EM164,DH164,DT164,DF164
1305  003300  070105
1306                                            ;ITEM 165
1307  003302  055224  066557  071142                .WORD   EM165,DH165,DT165,DF165
1308  003310  070265
1309                                            ;ITEM 166
1310  003312  055310  066557  071142                .WORD   EM166,DH166,DT166,DF166
1311  003320  070265
1312                                            ;ITEM 167
1313  003322  055373  066557  071142                .WORD   EM167,DH167,DT167,DF167
1314  003330  070265
1315                                            ;ITEM 170
1316  003332  055425  066557  071142                .WORD   EM170,DH170,DT170,DF170
1317  003340  070265
1318                                            ;ITEM 171
1319  003342  055513  066557  071142                .WORD   EM171,DH171,DT171,DF171
1320  003350  070265
1321                                            ;ITEM 172
1322  003352  055636  066557  071142                .WORD   EM172,DH172,DT172,DF172
1323  003360  070265
1324                                            ;ITEM 173
1325  003362  055723  066557  071142                .WORD   EM173,DH173,DT173,DF173
1326  003370  070265
1327                                            ;ITEM 174
1328  003372  056071  066557  071142                .WORD   EM174,DH174,DT174,DF174
1329  003400  070265
1330                                            ;ITEM 175
1331  003402  056237  066557  071142                .WORD   EM175,DH175,DT175,DF175
1332  003410  070265
1333                                            ;ITEM 176
1334  003412  056351  066557  071142                .WORD   EM176,DH176,DT176,DF176
```

```
1335  003420  070317
1336                                  ;ITEM 177
1337  003422  056435  066557  071142          .WORD   EM177,DH177,DT177,DF177
1338  003430  070317
1339                                  ;ITEM 200
1340  003432  056520  066557  071142          .WORD   EM200,DH200,DT200,DF200
1341  003440  070317
1342                                  ;ITEM 201
1343  003442  056552  066557  071142          .WORD   EM201,DH201,DT201,DF201
1344  003450  070317
1345                                  ;ITEM 202
1346  003452  056641  066557  071142          .WORD   EM202,DH202,DT202,DF202
1347  003460  070317
1348                                  ;ITEM 203
1349  003462  057003  066557  071142          .WORD   EM203,DH203,DT203,DF203
1350  003470  070317
1351                                  ;ITEM 204
1352  003472  057145  066557  071142          .WORD   EM204,DH204,DT204,DF204
1353  003500  070317
1354                                  ;ITEM 205
1355  003502  057233  066557  071142          .WORD   EM205,DH205,DT205,DF205
1356  003510  070317
1357                                  ;ITEM 206
1358  003512  057401  066557  071142          .WORD   EM206,DH206,DT206,DF206
1359  003520  070317
1360                                  ;ITEM 207
1361  003522  057547  066716  071252          .WORD   EM207,DH207,DT207,DF207
1362  003530  070361
1363                                  ;ITEM 210
1364  003532  057611  067006  071230          .WORD   EM210,DH210,DT210,DF210
1365  003540  070351
1366                                  ;ITEM 211
1367  003542  057653  067006  071252          .WORD   EM211,DH211,DT211,DF211
1368  003550  070361
1369                                  ;ITEM 212
1370  003552  060017  066716  071324          .WORD   EM212,DH212,DT212,DF212
1371  003560  070405
1372                                  ;ITEM 213
1373  003562  060203  066716  071324          .WORD   EM213,DH213,DT213,DF213
1374  003570  070405
1375                                  ;ITEM 214
1376  003572  060367  066716  071324          .WORD   EM214,DH214,DT214,DF214
1377  003600  070405
1378                                  ;ITEM 215
1379  003602  060553  066716  071324          .WORD   EM215,DH215,DT215,DF215
1380  003610  070405
1381                                  ;ITEM 216
1382  003612  060737  067006  071252          .WORD   EM216,DH216,DT216,DF216
1383  003620  070361
1384                                  ;ITEM 217
1385  003622  061121  067047  071406          .WORD   EM217,DH217,DT217,DF217
1386  003630  070435
1387                                  ;ITEM 220
1388  003632  061163  066656  071252          .WORD   EM220,DH220,DT220,DF220
1389  003640  070361
1390                                  ;ITEM 221
```

```
1391  003642  061413  067006  071252          .WORD   EM221,DH221,DT221,DF221
1392  003650  070361
1393                          ;ITEM 222
1394  003652  061657  066656  071252          .WORD   EM222,DH222,DT222,DF222
1395  003660  070361
1396                          ;ITEM 223
1397  003662  062110  067006  071252          .WORD   EM223,DH223,DT223,DF223
1398  003670  070361
1399                          ;ITEM 224
1400  003672  062355  066656  071252          .WORD   EM224,DH224,DT224,DF224
1401  003700  070361
1402                          ;ITEM 225
1403  003702  062606  067006  071252          .WORD   EM225,DH225,DT225,DF225
1404  003710  070361
1405                          ;ITEM 226
1406  003712  063053  067122  071324          .WORD   EM226,DH226,DT226,DF226
1407  003720  070405
1408                          ;ITEM 227
1409  003722  063206  067211  071324          .WORD   EM227,DH227,DT227,DF227
1410  003730  070405
1411                          ;ITEM 230
1412  003732  063341  067122  071324          .WORD   EM230,DH230,DT230,DF230
1413  003740  070405
1414                          ;ITEM 231
1415  003742  063475  067211  071230          .WORD   EM231,DH231,DT231,DF231
1416  003750  070405
1417                          ;ITEM 232
1418  003752  057611  067211  071252          .WORD   EM232,DH232,DT232,DF232
1419  003760  070405
1420                          ;ITEM 233
1421  003762  063631  067300  071426          .WORD   EM233,DH233,DT233,DF233
1422  003770  070444
1423                          ;ITEM 234
1424  003772  063670  067341  071462          .WORD   EM234,DH234,DT234,DF234
1425  004000  070461
1426                          ;ITEM 235
1427  004002  063754  067427  071502          .WORD   EM235,DH235,DT235,DF235
1428  004010  070470
1429                          ;ITEM 236
1430  004012  064022  067467  071462          .WORD   EM236,DH236,DT236,DF236
1431  004020  070461
1432                          ;ITEM 237
1433  004022  064055  067341  071462          .WORD   EM237,DH237,DT237,DF237
1434  004030  070461
1435                          ;ITEM 240
1436  004032  064141  067555  071462          .WORD   EM240,DH240,DT240,DF240
1437  004040  070461
1438                          ;ITEM 241
1439  004042  064175  067300  071426          .WORD   EM241,DH241,DT241,DF241
1440  004050  070444
1441                          ;ITEM 242
1442  004052  064300  067555  071462          .WORD   EM242,DH242,DT242,DF242
1443  004060  070461
1444                          ;ITEM 243
1445  004062  064334  067300  071426          .WORD   EM243,DH243,DT243,DF243
1446  004070  070444
```

```
1447
1448  004072  064437  067300  071426          ;ITEM 244
1449  004100  070444                .WORD   EM244,DH244,DT244,DF244
1450
1451  004102  064476  067300  071426          ;ITEM 245
1452  004110  070444                .WORD   EM245,DH245,DT245,DF245
1453
1454  004112  043341  066216  070726          ;ITEM 246
1455  004120  070105                .WORD   EM246,DH246,DT246,DF246
1456
1457  004122  064560  067645  071514          ;ITEM 247
1458  004130  070474                .WORD   EM247,DH247,DT247,DF247
1459
1460  004132  064614  067712  071532          ;ITEM 250
1461  004140  070474                .WORD   EM250,DH250,DT250,DF250
1462
1463  004142  064646  067712  071532          ;ITEM 251
1464  004150  070474                .WORD   EM251,DH251,DT251,DF251
1465
1466  004152  064701  066306  071544          ;ITEM 252
1467  004160  070502                .WORD   EM252,DH252,DT252,DF252
1468
1469  004162  064772  066216  071556          ;ITEM 253
1470  004170  070506                .WORD   EM253,DH253,DT253,DF253
1471
1472  004172  065056  066216  071556          ;ITEM 254
1473  004200  070506                .WORD   EM254,DH254,DT254,DF254
1474
1475  004202  065143  066216  071556          ;ITEM 255
1476  004210  070506                .WORD   EM255,DH255,DT255,DF255
1477
1478  004212  065231  066216  071556          ;ITEM 256
1479  004220  070506                .WORD   EM256,DH256,DT256,DF256
1480
1481  004222  065320  066216  071556          ;ITEM 257
1482  004230  070506                .WORD   EM257,DH257,DT257,DF257
1483
1484  004232  065406  066216  071556          ;ITEM 260
1485  004240  070506                .WORD   EM260,DH260,DT260,DF260
1486
1487  004242  065475  066216  071556          ;ITEM 261
1488  004250  070506                .WORD   EM261,DH261,DT261,DF261
1489
1490  004252  065565  066216  071556          ;ITEM 262
1491  004260  070506                .WORD   EM262,DH262,DT262,DF262
1492
1493  004262  065656  066216  071556          ;ITEM 263
1494  004270  070506                .WORD   EM263,DH263,DT263,DF263
1495
1496  004272  065743  066216  071556          ;ITEM 264
1497  004300  070506                .WORD   EM264,DH264,DT264,DF264
1498
1499  004302  066030  066216  071556          ;ITEM 265
1500  004310  070506                .WORD   EM265,DH265,DT265,DF265
1501
1502  004312  066115  067752  071544          ;ITEM 266
                                  .WORD   EM266,DH266,DT266,DF266
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
004322                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  033362                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAC IN .SEOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
004322                                  .=$SVPC                 ;; RESTORE PC
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
004322                                  .$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  004322                          $APTHDR ;;POINT TO APT HEADER BLOCK
004322                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

004322                          $APTHD:
004322  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
004324  001316                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
004326  000010                  $TSTM:  .WORD   10      ;;RUN TIM OF LONGEST TEST
004330  000040                  $PASTM: .WORD   40      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
004332  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
004334  000052                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

004336                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004336  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004342  005026                  1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004344  022706  001140                  CMP     #$WR,R6 ;;DONE?
004350  001374                          BNE     1$-6            ;;LOOP BACK IF NO
004352  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
004356  012737  033442  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
004364  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
004372  012737  033722  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004400  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004406  012737  035670  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004414  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
004422  012737  035754  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
004430  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
004436  016767  026542  026532          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
004444  005067  174632                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
```

```
1559  004450  005067  174630                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1560  004454  112767  000001  174433          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1561                                  ;;INITIALIZE THE "T-BIT" TRAP VECTOR, THEN LOAD LOCATION "$RTRN", IN
1562                                  ;;THE "END-OF-PASS" ($EOP) ROUTINE, WITH A "RTI" OR "RTT".
1563  004462  012737  033426  000014          MOV     #$RTRN,@#TBITVEC ;;SET "T" BIT VECTOR TO $RTRN
1564  004470  012737  000340  000016          MOV     #340,@#TBITVEC+2 ;;LEVEL 7
1565  004476  012767  000002  026722          MOV     #RTI,$RTRN      ;;SET $RTRN TO A RTI
1566  004504  012737  004532  000010          MOV     #65$,@#RESVEC   ;;TRY TO DO A RTT
1567  004512  005046                          CLR     -(SP)           ;;DUMMY PS
1568  004514  012746  004522                  MOV     #64$,-(SP)      ;;AND PC
1569  004520  000006                          RTT                     ;;TRY THE RTT
1570  004522  012767  000006  026676  64$:    MOV     #RTT,$RTRN      ;;RTT IS LEGAL--SET $RTRN TO A RTT
1571  004530  000402                          BR      66$
1572  004532  062706  000010          65$:    ADD     #10,SP          ;;RTT ILLEGAL--CLEAN OFF THE STACK
1573  004536  012737  000012  000010  66$:    MOV     #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
1574  004544  005067  026664                  CLR     $TBIT           ;;CLEAR "T" BIT SWITCH
1575  004550  012767  004550  174330          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1576  004556  012767  004556  174324          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1577                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1578                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1579  004564  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1580  004570  012737  004624  000004          MOV     #67$,@#ERRVEC   ;;SET UP ERROR VECTOR
1581  004576  012767  177570  174334          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1582  004604  012767  177570  174330          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1583  004612  022777  177777  174320          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1584  004620  001012                          BNE     69$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1585                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1586  004622  000403                          BR      68$             ;;BRANCH IF NO TIMEOUT
1587  004624  012716  004632          67$:    MOV     #68$,(SP)       ;;SET UP FOR TRAP RETURN
1588  004630  000002                          RTI
1589  004632  012767  000176  174300  68$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1590  004640  012767  000174  174274          MOV     #DISPREG,DISPLAY
1591  004646  012637  000004          69$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1592
1593  004652  005067  174446                  CLR     $PASS           ;;CLEAR PASS COUNT
1594  004656  132767  000200  174453          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1595  004664  001403                          BEQ     70$             ;;YES,USE NON-APT SWITCH
1596  004666  012767  001340  174244          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1597  004674                          70$:
1598                                  .SBTTL  TYPE PROGRAM NAME
1599                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1600  004674  005227  177777                  INC     #-1             ;;FIRST TIME?
1601  004700  001055                          BNE     71$             ;;BRANCH IF NO
1602  004702  022737  033362  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
1603  004710  001451                          BEQ     71$             ;;BRANCH IF YES
1604  004712  104401  004760                  TYPE    ,72$            ;;TYPE ASCIZ STRING
1605                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1606  004716  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1607  004722  001012                          BNE     73$             ;;BRANCH IF YES
1608  004724  126727  174406  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1609  004732  001406                          BEQ     73$             ;;BRANCH IF YES
1610  004734  026727  174200  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1611  004742  001005                          BNE     74$             ;;BRANCH IF NO
1612  004744  104405                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1613  004746  000403                          BR      74$
1614  004750  112767  000001  174156  73$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
```

```
                                74$:
                                        BR      71$             ;;GET OVER THE ASCIZ
                                ;;72$:  .ASCIZ  <CRLF>*DFFPB, FPII-A 11/34 FPP DIAGNOSTIC PART 2* <CRLF
                                71$:
                                LOOP:


                                ;;*******************************************************************
                                ;*TEST 1         ROUND\TRUNK TEST
                                ;*
                                ;* THIS IS A TEST OF THE ROUND\TRUNK
                                ;* FLOWS.  IN PARTICULAR TWO THINGS ARE TESTED:
                                ;* FIRST A CONDITION IN WHICH ROUNDING
                                ;* RESULTS IN THE NEED FOR RENORMALIZATION, AND
                                ;* SECOND  THE PSW CONDITION CODES N AND
                                ;* Z BIT COMMBINATIONS
                                ;*
                                ;;*******************************************************************
005034  000004                TST1:   SCOPE

                                        ;ROUND AND NORMALIZE TEST

005036                          HH1:
005036  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
005040  012704  003200                  MOV     #3200,R4                ;SET FIU, FIV, AND FD
005044  170104                          LDFPS   R4
005046  012737  005066  001236          MOV     #HH2,@#$TMP2
005054  012700  006616                  MOV     #HHP0,R0                ;SET ACO OPERAND
005060  172410                          LDD     (R0),AC0
005062  012700  006626                  MOV     #HHP1,R0                ;FSPC
005066  172010                  HH2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
005070  170205                          STFPS   R5                      ;GET FPS
005072  012700  006606                  MOV     #HHDAT0,R0              ;GET THE RESULT
005076  174010                          STD     AC0,(R0)
005100  012701  006636                  MOV     #HHP2,R1                ;IS IT CORRECT
005104  012702  000004                  MOV     #4,R2
005110  022021                  HH3:    CMP     (R0)+,(R1)+
005112  001415                          BEQ     HH6
005114  012700  006606                  MOV     #HHDAT0,R0              ;DID FLOW GO
005120  012701  006646                  MOV     #HHP3,R1                ;FROM STATE 663
005124  012702  000004                  MOV     #4,R2                   ;TO 313 INSTEAD
005130  022021                  HH4:    CMP     (R0)+,(R1)+             ;OF TO 353
005132  001402                          BEQ     HH5
005134  000137  005626                  JMP     @#HHERO
005140  077205                  HH5:    SOB     R2,HH4
005142  000137  005674                  JMP     @#HHER1
005146  077220                  HH6:    SOB     R2,HH3
005150  020405                          CMP     R4,R5           ;FPS CORRECT?
005152  001402                          BEQ     HH7
005154  000137  005742                  JMP     @#HHER00

                                ;THIS IS A TEST OF THE ABILITY
                                ;OF NORMALIZE TO PRODUCE A ZERO EXP. AND
                                ;OF THE R\T ALGORITHM TO PORPERLY SET THE FPS
```

```
1671  005160                          HH7:
1672  005160  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
1673  005162  012704  043200                  MOV     #043200,R4      ;SET FIU,FIV,AND FD
1674                                                                    ;FID
1675  005166  170104                          LDFPS   R4
1676  005170  012737  005216  001236          MOV     #HH8,@#$TMP2    ;IN CASE UNDERFLOW
1677  005176  012737  006536  000244          MOV     #HHTRAP,@#FPVECT ;TRAP OCCURS
1678  005204  012700  006666                  MOV     #HHP5,R0        ;SET ACO OPERAND
1679  005210  172410                          LDD     (R0),AC0
1680  005212  012700  006676                  MOV     #HHP6,R0        ;FSPC
1681  005216  172010                  HH8:    ADDD    (R0),AC0        ;TEST INSTRUCTION
1682  005220  170205                          STFPS   R5              ;GET FPS
1683  005222  012700  006606                  MOV     #HHDAT0,R0      ;GET THE RESULT
1684  005226  174010                          STD     AC0,(R0)
1685  005230  012701  006656                  MOV     #HHP4,R1        ;IS IT CORRECT
1686  005234  012702  000004                  MOV     #4,R2
1687  005240  022021                  HH9:    CMP     (R0)+,(R1)+
1688  005242  001402                          BEQ     HH10
1689  005244  000137  006010                  JMP     @#HHER2
1690  005250  077205                  HH10:   SOB     R2,HH9
1691  005252  052704  100004                  BIS     #100004,R4      ;FPS CORRECT?
1692  005256  020405                          CMP     R4,R5
1693  005260  001402                          BEQ     HH11
1694  005262  000137  006056                  JMP     @#HHER3
1695                                  ;THIS IS A TEST OF THE R\T ALGORITHM'S
1696                                  ;ABILITY TO SET BOTH N AND Z ON A - 0 RESULT.
1697  005266                          HH11:
1698  005266  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
1699                                                                    ;SET FIV, FIV, AND FD
1700  005270  012704  043200                  MOV     #043200,R4
1701  005274  170104                          LDFPS   R4
1702  005276  012737  005316  001236          MOV     #HH12,@#$TMP2
1703  005304  012700  006716                  MOV     #HHP8,R0        ;SET ACO OPERAND
1704  005310  172410                          LDD     (R0),AC0
1705  005312  012700  006726                  MOV     #HHP9,R0        ;FSPC
1706  005316  172010                  HH12:   ADDD    (R0),AC0        ;TEST INSTRUCTION
1707  005320  170205                          STFPS   R5              ;GET FPS
1708  005322  012700  006606                  MOV     #HHDAT0,R0      ;GET THE RESULT
1709  005326  174010                          STD     AC0,(R0)
1710  005330  012701  006706                  MOV     #HHP7,R1        ;IS IT CORRECT
1711  005334  012702  000004                  MOV     #4,R2
1712  005340  022021                  HH13:   CMP     (R0)+,(R1)+
1713  005342  001415                          BEQ     HH16
1714  005344  012700  006606                  MOV     #HHDAT0,R0
1715  005350  012701  006656                  MOV     #HHP4,R1
1716  005354  012702  000004                  MOV     #4,R2
1717  005360  022021                  HH14:   CMP     (R0)+,(R1)+
1718  005362  001402                          BEQ     HH15
1719  005364  000137  006124                  JMP     @#HHER4
1720  005370  077205                  HH15:   SOB     R2,HH14
1721  005372  000137  006172                  JMP     @#HHER5
1722  005376  077220                  HH16:   SOB     R2,HH13
1723  005400  052704  100014                  BIS     #100014,R4      ;FPS CORRECT?
1724  005404  020405                          CMP     R4,R5
1725  005406  001402                          BEQ     HH17
1726  005410  000137  006240                  JMP     @#HHER6
```

```
                                        ;TEST THAT CC ARE CLEARED BY R\T
005414                                  HH17:
005414  104413                                LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
005416  012704  000200                        MOV     #00200,R4               ;SET FIV, FIV, AND FD
005422  170104                                LDFPS   R4
005424  012737  005452  001236                MOV     #HH18,@#$TMP2
005432  012737  036614  000244                MOV     #FPSPUR,@#FPVECT
005440  012700  006716                        MOV     #HHP8,R0                ;SET ACO OPERAND
005444  172410                                LDD     (R0),AC0
005446  012700  006716                        MOV     #HHP8,R0                ;FSPC
005452  172010                          HH18: ADDD    (R0),AC0                ;TEST INSTRUCTION
005454  170205                                STFPS   R5                      ;GET FPS
005456  012700  006606                        MOV     #HHDAT0,R0              ;GET THE RESULT
005462  174010                                STD     AC0,(R0)
005464  012701  006736                        MOV     #HHP10,R1               ;IS IT CORRECT
005470  012702  000004                        MOV     #4,R2
005474  022021                          HH19: CMP     (R0)+,(R1)+
005476  001402                                BEQ     HH20
005500  000137  006306                        JMP     @#HHER7
005504  077205                          HH20: SOB     R2,HH19
005506  052704  000000                        BIS     #00000,R4               ;FPS CORRECT?
005512  020405                                CMP     R4,R5
005514  001402                                BEQ     HH21
005516  000137  006354                        JMP     @#HHER8
                                        ;TEST THAT N IS SET BY R\T
005522                                  HH21:
005522  104413                                LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
005524  012704  003200                        MOV     #3200,R4                ;SET FIV, FIV, AND FD
005530  170104                                LDFPS   R4
005532  012737  005552  001236                MOV     #HH22,@#$TMP2
005540  012700  006666                        MOV     #HHPS,R0                ;SET ACO OPERAND
005544  172410                                LDD     (R0),AC0
005546  012700  006666                        MOV     #HHPS,R0                ;FSPC
005552  172010                          HH22: ADDD    (R0),AC0                ;TEST INSTRUCTION
005554  170205                                STFPS   R5                      ;GET FPS
005556  012700  006606                        MOV     #HHDAT0,R0              ;GET THE RESULT
005562  174010                                STD     AC0,(R0)
005564  012701  006746                        MOV     #HHP11,R1               ;IS IT CORRECT
005570  012702  000004                        MOV     #4,R2
005574  022021                          HH23: CMP     (R0)+,(R1)+
005576  001402                                BEQ     HH24
005600  000137  006422                        JMP     @#HHER9
005604  077205                          HH24: SOB     R2,HH23
005606  052704  000010                        BIS     #10,R4
005612  020405                                CMP     R4,R5                   ;FPS CORRECT?
005614  001402                                BEQ     HH25
005616  000137  006470                        JMP     @#HHER10
005622  000137  006756                  HH25: JMP     @#HHDONE
005626                                  HHERO:
005626  010537  001252                        MOV     R5,@#$TMP10
005632  010437  001254                        MOV     R4,@#$TMP11
005636  012737  006626  001240                MOV     #HHP1,@#$TMP3
005644  012737  006616  001242                MOV     #HHP0,@#$TMP4
005652  012737  006606  001244                MOV     #HHDAT0,@#$TMP5
005660  012737  006636  001246                MOV     #HHP2,@#$TMP6
005666  104207                          1$:   ERROR   207
```

```
1783  005670  000137  006756              JMP     @#HHDONE
1784  005674                      HHER1:
1785  005674  010537  001252              MOV     R5,@#$TMP10
1786  005700  010437  001254              MOV     R4,@#$TMP11
1787  005704  012737  006626  001240      MOV     #HHP1,@#$TMP3
1788  005712  012737  006616  001242      MOV     #HHP0,@#$TMP4
1789  005720  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1790  005726  012737  006636  001246      MOV     #HHP2,@#$TMP6
1791  005734  104211              1$:     ERROR   211
1792  005736  000137  006756              JMP     @#HHDONE
1793  005742                      HHER00:
1794  005742  010537  001252              MOV     R5,@#$TMP10
1795  005746  010437  001254              MOV     R4,@#$TMP11
1796  005752  012737  006626  001240      MOV     #HHP1,@#$TMP3
1797  005760  012737  006616  001242      MOV     #HHP0,@#$TMP4
1798  005766  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1799  005774  012737  006636  001246      MOV     #HHP2,@#$TMP6
1800  006002  104210              1$:     ERROR   210
1801  006004  000137  006756              JMP     @#HHDONE
1802  006010                      HHER2:
1803  006010  010537  001252              MOV     R5,@#$TMP10
1804  006014  010437  001254              MOV     R4,@#$TMP11
1805  006020  012737  006676  001240      MOV     #HHP6,@#$TMP3
1806  006026  012737  006666  001242      MOV     #HHP5,@#$TMP4
1807  006034  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1808  006042  012737  006656  001246      MOV     #HHP4,@#$TMP6
1809  006050  104207              1$:     ERROR   207
1810  006052  000137  006756              JMP     @#HHDONE
1811  006056                      HHER3:
1812  006056  010537  001252              MOV     R5,@#$TMP10
1813  006062  010437  001254              MOV     R4,@#$TMP11
1814  006066  012737  006676  001240      MOV     #HHP6,@#$TMP3
1815  006074  012737  006666  001242      MOV     #HHP5,@#$TMP4
1816  006102  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1817  006110  012737  006656  001246      MOV     #HHP4,@#$TMP6
1818  006116  104214              1$:     ERROR   214
1819  006120  000137  006756              JMP     @#HHDONE
1820  006124                      HHER4:
1821  006124  010537  001252              MOV     R5,@#$TMP10
1822  006130  010437  001254              MOV     R4,@#$TMP11
1823  006134  012737  006726  001240      MOV     #HHP9,@#$TMP3
1824  006142  012737  006716  001242      MOV     #HHP8,@#$TMP4
1825  006150  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1826  006156  012737  006706  001246      MOV     #HHP7,@#$TMP6
1827  006164  104207              1$:     ERROR   207
1828  006166  000137  006756              JMP     @#HHDONE
1829  006172                      HHER5:
1830  006172  010537  001252              MOV     R5,@#$TMP10
1831  006176  010437  001254              MOV     R4,@#$TMP11
1832  006202  012737  006726  001240      MOV     #HHP9,@#$TMP3
1833  006210  012737  006716  001242      MOV     #HHP8,@#$TMP4
1834  006216  012737  006606  001244      MOV     #HHDAT0,@#$TMP5
1835  006224  012737  006706  001246      MOV     #HHP7,@#$TMP6
1836  006232  104216              1$:     ERROR   216
1837  006234  000137  006756              JMP     @#HHDONE
1838  006240                      HHER6:
```

```
1839  006240  010537  001252                  MOV     R5,@#$TMP10
1840  006244  010437  001254                  MOV     R4,@#$TMP11
1841  006250  012737  006726  001240          MOV     #HHP9,@#$TMP3
1842  006256  012737  006716  001242          MOV     #HHP8,@#$TMP4
1843  006264  012737  006606  001244          MOV     #HHDAT0,@#$TMP5
1844  006272  012737  006706  001246          MOV     #HHP7,@#$TMP6
1845  006300  104215                  1$:     ERROR   215
1846  006302  000137  006756                  JMP     @#HHDONE
1847  006306                          HHER7:
1848  006306  010537  001252                  MOV     R5,@#$TMP10
1849  006312  010437  001254                  MOV     R4,@#$TMP11
1850  006316  012737  006716  001240          MOV     #HHP8,@#$TMP3
1851  006324  012737  006716  001242          MOV     #HHP8,@#$TMP4
1852  006332  012737  006606  001244          MOV     #HHDAT0,@#$TMP5
1853  006340  012737  006736  001246          MOV     #HHP10,@#$TMP6
1854  006346  104207                  1$:     ERROR   207
1855  006350  000137  006756                  JMP     @#HHDONE
1856  006354                          HHER8:
1857  006354  010537  001252                  MOV     R5,@#$TMP10
1858  006360  010437  001254                  MOV     R4,@#$TMP11
1859  006364  012737  006716  001240          MOV     #HHP8,@#$TMP3
1860  006372  012737  006716  001242          MOV     #HHP8,@#$TMP4
1861  006400  012737  006606  001244          MOV     #HHDAT0,@#$TMP5
1862  006406  012737  006736  001246          MOV     #HHP10,@#$TMP6
1863  006414  104212                  1$:     ERROR   212
1864  006416  000137  006756                  JMP     @#HHDONE
1865  006422                          HHER9:
1866  006422  010537  001252                  MOV     R5,@#$TMP10
1867  006426  010437  001254                  MOV     R4,@#$TMP11
1868  006432  012737  006666  001240          MOV     #HHPS,@#$TMP3
1869  006440  012737  006666  001242          MOV     #HHPS,@#$TMP4
1870  006446  012737  006606  001244          MOV     #HHDAT0,@#$TMP5
1871  006454  012737  006746  001246          MOV     #HHP11,@#$TMP6
1872  006462  104207                  1$:     ERROR   207
1873  006464  000137  006756                  JMP     @#HHDONE
1874  006470                          HHER10:
1875  006470  010537  001252                  MOV     R5,@#$TMP10
1876  006474  010437  001254                  MOV     R4,@#$TMP11
1877  006500  012737  006666  001240          MOV     #HHPS,@#$TMP3
1878  006506  012737  006666  001242          MOV     #HHPS,@#$TMP4
1879  006514  012737  006606  001244          MOV     #HHDAT0,@#$TMP5
1880  006522  012737  006746  001246          MOV     #HHP11,@#$TMP6
1881  006530  104213                  1$:     ERROR   213
1882  006532  000137  006756                  JMP     @#HHDONE
1883  006536  013703  001236          HHTRAP: MOV     @#$TMP2,R3      ;WAS THE TRAP TO 244
1884  006542  062703  000002                  ADD     #2,R3           ;ON THE INSTRUCTION
1885  006546  020316                          CMP     R3,(SP)         ;BEING TESTED?
1886  006550  001402                          BEQ     1$
1887  006552  000137  036614                  JMP     @#FPSPUR
1888  006556  011637  001236          1$:     MOV     (SP),@#$TMP2    ;FAILURE OF FPS INTERRUPT
1889                                                                  ;DISABLE BIT (FID=1)
1890  006562  022626                          CMP     (SP)+,(SP)+     ;TO INHIBIT TRAP.
1891  006564  170201                          STFPS   R1
1892  006566  010137  001240                  MOV     R1,@#$TMP3
1893  006572  170301                          STST    R1
1894  006574  010137  001242                  MOV     R1,@#$TMP4
```

```
1895  006600  104217                  2$:     ERROR   217
1896  006602  000137  006756          JMP     @#HHDONE
1897  006606  000000                  HHDAT0: 0
1898  006610  000000                          0
1899  006612  000000                          0
1900  006614  000000                          0
1901  006616  000452                  HHP0:   452
1902  006620  125252                          125252
1903  006622  125252                          125252
1904  006624  125253                          125253
1905  006626  000252                  HHP1:   252
1906  006630  125252                          125252
1907  006632  125252                          125252
1908  006634  125252                          125252
1909  006636  000600                  HHP2:   600                     ;HHP0 + HHP1 WITH
1910  006640  000000                          0                       ;PROPER NORMALIZATION
1911  006642  000000                          0
1912  006644  000000                          0
1913  006646  000400                  HHP3:   400                     ;HHP0 + HHP1 WITH
1914  006650  000000                          0                       ;BAD NORMALIZATION
1915  006652  000000                          0
1916  006654  000000                          0
1917  006656  000000                  HHP4:   0
1918  006660  000000                          0
1919  006662  000000                          0
1920  006664  000000                          0
1921  006666  100200                  HHP5:   100200
1922  006670  000000                          0
1923  006672  000000                          0
1924  006674  000000                          0
1925  006676  000300                  HHP6:   300
1926  006700  000000                          0
1927  006702  000000                          0
1928  006704  000000                          0
1929  006706  100000                  HHP7:   100000                  ;HHP7 = HHP8 + HHP9
1930  006710  000000                          0                       ;     = HHP5 + HHP6
1931  006712  000000                          0
1932  006714  000000                          0
1933  006716  000200                  HHP8:   200
1934  006720  000000                          0
1935  006722  000000                          0
1936  006724  000000                          0
1937  006726  100300                  HHP9:   100300
1938  006730  000000                          0
1939  006732  000000                          0
1940  006734  000000                          0
1941  006736  000400                  HHP10:  400                     ;HHP10 = HHP8 + HHP8
1942  006740  000000                          0
1943  006742  000000                          0
1944  006744  000000                          0
1945  006746  100400                  HHP11:  100400                  ;HHP11 = HHP5 + HHP5
1946  006750  000000                          0
1947  006752  000000                          0
1948  006754  000000                          0
1949  006756                          HHDONE:
1950  006756  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
```

```
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER  WAS
                                                        ;THE USER TYPED CONTROL G?).


                                        ;;**************************************************************
                                        ;*TEST 2        OVER\UNDER TEST
                                        ;*
                                        ;*THIS IS A PARTIAL TEST OF THE OVER\UNDER
                                        ;*FLOWS.  ONE OVERFLOW AND TWO UNDERFLOW
                                        ;*CONDITIONS ARE CHECKED.  THE REMAINING
                                        ;*UNDERFLOW COND. AND THE REMAINING OVERFLOW
                                        ;*COND. WILL BE CHECKED LATER USING THE
                                        ;*XXX INSTRUCTION.  HERE EACH CONDITION TESTED
                                        ;*IS CHECKED BOTH WITH TRAPS ENABLED
                                        ;*(FIU=1 OR FIV=1) AND ALSO WITH TRAPS
                                        ;*DISABLED (FIU=0 OR FIV=0).
                                        ;*
                                        ;;**************************************************************
006760  000004                        TST2:   SCOPE

                                        ;TEST OVERFLOW CONDITION WITH TRAP DISABLER FIV=0
006762                                  GG1:
006762  104413                                LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
006764  012704  000200                        MOV     #200,R4         ;CLEAR FIU, FIV, AND SET FD
006770  170104                                LDFPS   R4
006772  012737  007020  001236                MOV     #GG2,@#$TMP2
007000  012737  010052  000244                MOV     #GGERR0,@#FPVECT
007006  012700  011624                        MOV     #GGP5,R0        ;SET AC0 OPERAND
007012  172410                                LDD     (R0),AC0
007014  012700  011624                        MOV     #GGP5,R0        ;FSRC
007020  172010                        GG2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
007022  170205                                STFPS   R5              ;GET FPS
007024  012700  011554                        MOV     #GGOUT0,R0      ;GET THE RESULT
007030  174010                                STD     AC0,(R0)
007032  012701  011634                        MOV     #GGP6,R1        ;IS IT CORRECT
007036  012702  000004                        MOV     #4,R2
007042  022021                        GG3:    CMP     (R0)+,(R1)+
007044  001402                                BEQ     GG4
007046  000137  010150                        JMP     @#GGER1
007052  077205                        GG4:    SOB     R2,GG3
007054  052704  000006                        BIS     #6,R4           ;FPS CORRECT?
007060  020405                                CMP     R4,R5
007062  001402                                BEQ     GG5
007064  000137  010216                        JMP     @#GGER2
                                        ;TEST OVERFLOW WITH TRAPS ENABLED
                                        ;FIV = 1
007070                                  GG5:
007070  104413                                LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
007072  012704  001200                        MOV     #1200,R4        ;CLEAR FIU, SET FIV, AND FD
007076  170104                                LDFPS   R4
007100  012737  007126  001236                MOV     #GG6,@#$TMP2
007106  012737  007144  000244                MOV     #GG7,@#FPVECT
007114  012700  011624                        MOV     #GGP5,R0        ;SET AC0 OPERAND
007120  172410                                LDD     (R0),AC0
```

```
2007  007122  012700  011624            MOV     #GGPS,R0        ;FSPC
2008  007126  172010            GG6:    ADDD    (R0),AC0        ;TEST INSTRUCTION
2009  007130  170000                    CFCC
2010  007132  012700  011554            MOV     #GGDAT0,R0
2011  007136  174010                    STD     AC0,(R0)
2012  007140  000137  010264            JMP     @#GGER3
2013  007144  013703  001236    GG7:    MOV     @#$TMP2,R3
2014  007150  062703  000002            ADD     #2,R3
2015  007154  020316                    CMP     R3,(SP)
2016  007156  001402                    BEQ     1$
2017  007160  000137  036614            JMP     @#FPSPUR
2018  007164  011637  001236    1$:     MOV     (SP),@#$TMP2
2019  007170  022626                    CMP     (SP)+,(SP)+
2020  007172  170205                    STFPS   R5
2021  007174  012700  011554            MOV     #GGDAT0,R0      ;GET THE RESULT
2022  007200  174010                    STD     AC0,(R0)
2023  007202  012701  011634            MOV     #GGP6,R1        ;IS IT CORRECT
2024  007206  012702  000004            MOV     #4,R2
2025  007212  022021            GG8:    CMP     (R0)+,(R1)+
2026  007214  001402                    BEQ     GG9
2027  007216  000137  010332            JMP     @#GGER4
2028  007222  077205            GG9:    SOB     R2,GG8
2029  007224  052704  100006            BIS     #100006,R4
2030  007230  020405                    CMP     R4,R5           ;FPS CORRECT?
2031  007232  001402                    BEQ     1$
2032  007234  000137  010402            JMP     @#GGER6
2033  007240  012704  000010    1$:     MOV     #10,R4
2034                            ;CHECK FEC
2035  007244  170305                    STST    R5
2036  007246  020405                    CMP     R4,R5
2037  007250  001402                    BEQ     GG10
2038  007252  000137  010334            JMP     @#GGER5
2039                            ;CHECK UNDER FLOW CONDITION WITH
2040                            ;TRAPS DISABLED (FIU = 0)
2041  007256                    GG10:
2042  007256  104413                    LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2043  007260  012704  000200            MOV     #0200,R4        ;SET FIU, FIV, AND FD
2044  007264  170104                    LDFPS   R4
2045  007266  012737  007314  001236    MOV     #GG11,@#$TMP2
2046  007274  012737  010450  000244    MOV     #GGER7,@#FPVECT
2047  007302  012700  011574            MOV     #GGP2,R0        ;SET AC0 OPERAND
2048  007306  172410                    LDD     (R0),AC0        ;FSRC
2049  007310  012700  011604            MOV     #GGP3,R0
2050  007314  172010            GG11:   ADDD    (R0),AC0        ;TEST INSTRUCTION
2051  007316  170205                    STFPS   R5              ;GET FPS
2052  007320  012700  011554            MOV     #GGDAT0,R0      ;GET THE RESULT
2053  007324  174010                    STD     AC0,(R0)
2054  007326  012701  011634            MOV     #GGP6,R1        ;IS IT CORRECT
2055  007332  012702  000004            MOV     #4,R2
2056  007336  022021            GG12:   CMP     (R0)+,(R1)+
2057  007340  001402                    BEQ     GG13
2058  007342  000137  010546            JMP     @#GGER8
2059  007346  077205            GG13:   SOB     R2,GG12
2060  007350  052704  000004            BIS     #4,R4           ;FPS CORRECT?
2061  007354  020405                    CMP     R4,R5
2062  007356  001402                    BEQ     GG14
```

```
2063  007360  000137  010614                JMP     @#GGER9
2064                                        ;CHECK UNDERFLOW CONDITION WITH
2065                                        ;TRAP ENABLED (FIU = 1)
2066  007364                        GG14:
2067  007364  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2068  007366  012704  002200                MOV     #2200,R4        ;SET FIU, FIV, AND FD
2069  007372  170104                        LDFPS   R4
2070  007374  012737  007422  001236        MOV     #GG15,@#$TMP2
2071  007402  012737  007440  000244        MOV     #GG16,@#FPVECT
2072  007410  012700  011574                MOV     #GGP2,R0        ;SET AC0 OPERAND
2073  007414  172410                        LDD     (R0),AC0        ;FSPC
2074  007416  012700  011604                MOV     #GGP3,R0
2075  007422  172010                GG15:   ADDD    (R0),AC0        ;TEST INSTRUCTION
2076  007424  170000                        CFCC
2077  007426  012700  011554                MOV     #GGDAT0,R0
2078  007432  174010                        STD     AC0,(R0)
2079  007434  000137  010662                JMP     @#GGER10
2080  007440  013703  001236        GG16:   MOV     @#$TMP2,R3
2081  007444  062703  000002                ADD     #2,R3
2082  007450  021603                        CMP     (SP),R3
2083  007452  001402                        BEQ     1$
2084  007454  000137  036614                JMP     @#FPSPUR
2085  007460  011637  001236        1$:     MOV     (SP),@#$TMP2
2086  007464  022626                        CMP     (SP)+,(SP)+
2087  007466  170205                        STFPS   R5              ;GET FPS
2088  007470  012700  011554                MOV     #GGDAT0,R0      ;GET THE RESULT
2089  007474  174010                        STD     AC0,(R0)
2090  007476  012701  011644                MOV     #GGP7,R1        ;IS IT CORRECT
2091  007502  012702  000004                MOV     #4,R2
2092  007506  022021                GG17:   CMP     (R0)+,(R1)+
2093  007510  001402                        BEQ     GG18
2094  007512  000137  010730                JMP     @#GGER11
2095  007516  077205                GG18:   SOB     R2,GG17
2096  007520  052704  100000                BIS     #100000,R4
2097  007524  020405                        CMP     R4,R5           ;FPS CORRECT?
2098  007526  001402                        BEQ     2$
2099  007530  000137  010776                JMP     @#GGER12
2100  007534                        2$:
2101  007534  012704  000012        1$:     MOV     #12,R4
2102                                        ;CHECK FEC
2103  007540  170305                        STST    R5
2104  007542  020405                        CMP     R4,R5
2105  007544  001402                        BEQ     GG19
2106  007546  000177  001272                JMP     @GGER13
2107                                        ;CHECK UNDERFLOW CONDITION WITH TRAPS
2108                                        ;DISABLED (FIU = 0)
2109  007552                        GG19:
2110  007552  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2111  007554  012704  000200                MOV     #0200,R4        ;SET FIU, FIV, AND FD
2112  007560  170104                        LDFPS   R4
2113  007562  012737  007610  001236        MOV     #GG20,@#$TMP2
2114  007570  012737  011112  000244        MOV     #GGER14,@#FPVECT
2115  007576  012700  011574                MOV     #GGP2,R0        ;SET AC0 OPERAND
2116  007602  172410                        LDD     (R0),AC0
2117  007604  012700  011654                MOV     #GGP8,R0        ;FSPC
2118  007610  172010                GG20:   ADDD    (R0),AC0        ;TEST INSTRUCTION
```

```
2119  007612  170205                          STFPS   R5              ;GET FPS
2120  007614  012700  011554                  MOV     #GGDAT0,R0      ;GET THE RESULT
2121  007620  174010                          STD     AC0,(R0)
2122  007622  012701  011634                  MOV     #GGP6,R1        ;IS IT CORRECT
2123  007626  012702  000004                  MOV     #4,R2
2124  007632  022021                  GG21:   CMP     (R0)+,(R1)+
2125  007634  001402                          BEQ     GG22
2126  007636  000137  011210                  JMP     @#GGER15
2127  007642  077205                  GG22:   SOB     R2,GG21
2128  007644  052704  000004                  BIS     #4,R4           ;FPS CORRECT?
2129  007650  020405                          CMP     R4,R5
2130  007652  001402                          BEQ     GG23
2131  007654  000137  011256                  JMP     @#GGER16
2132                                          ;CHECK UNDERFLOW CONDITION WITH TRAP
2133                                          ;ENABLED (FIU = 1)
2134  007660                          GG23:
2135  007660  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2136  007662  012704  002200                  MOV     #2200,R4        ;SET FIU, FIV, AND FD
2137  007666  170104                          LDFPS   R4
2138  007670  012737  007716  001236          MOV     #GG24,@#$TMP2
2139  007676  012737  007734  000244          MOV     #GG25,@#FPVECT
2140  007704  012700  011574                  MOV     #GGP2,R0        ;SET ACO OPERAND
2141  007710  172410                          LDD     (R0),AC0
2142  007712  012700  011654                  MOV     #GGP8,R0        ;FSRC
2143  007716  172010                  GG24:   ADDD    (R0),AC0        ;TEST INSTRUCTION
2144  007720  170000                          CFCC
2145  007722  012700  011554                  MOV     #GGDAT0,R0
2146  007726  174010                          STD     AC0,(R0)
2147  007730  000137  011324                  JMP     @#GGER17
2148  007734  013700  001236          GG25:   MOV     @#$TMP2,R0
2149  007740  062700  000002                  ADD     #2,R0
2150  007744  020016                          CMP     R0,(SP)
2151  007746  001402                          BEQ     1$
2152  007750  000137  036614                  JMP     @#FPSPUR
2153  007754  011637  001236          1$:     MOV     (SP),@#$TMP2
2154  007760  022626                          CMP     (SP)+,(SP)+
2155  007762  170205                          STFPS   R5              ;GET FPS
2156  007764  012700  011554                  MOV     #GGDAT0,R0      ;GET THE RESULT
2157  007770  174010                          STD     AC0,(R0)
2158  007772  012701  011664                  MOV     #GGP9,R1        ;IS IT CORRECT
2159  007776  012702  000004                  MOV     #4,R2
2160  010002  022021                  GG26:   CMP     (R0)+,(R1)+
2161  010004  001402                          BEQ     GG27
2162  010006  000137  011372                  JMP     @#GGER18
2163  010012  077205                  GG27:   SOB     R2,GG26
2164  010014  052704  100004                  BIS     #100004,R4
2165  010020  020405                          CMP     R4,R5           ;FPS CORRECT?
2166  010022  001402                          BEQ     1$
2167  010024  000167  001456                  JMP     GGER20
2168  010030  012704  000012          1$:     MOV     #12,R4
2169                                          ;CHECK FEC
2170  010034  170305                          STST    R5
2171  010036  020405                          CMP     R4,R5
2172  010040  001402                          BEQ     GG28
2173  010042  000167  001372                  JMP     GGER19
2174
```

```
2175  010046  000137  011674          GG28:   JMP     @#GGDONE
2176
2177  010052  013701  001236          GGER0:  MOV     @#STMP2,R.
2178  010056  062701  000002                  ADD     #2,R1
2179  010062  020116                          CMP     R1,(SP)
2180  010064  001402                          BEQ     10$
2181  010066  000137  036614          5$:     JMP     @#FPSPUR
2182  010072                          10$:
2183  010072  170301                          STST    R1
2184  010074  020127  000010                  CMP     R1,#10
2185  010100  001372                          BNE     5$
2186  010102  022626                          CMP     (SP)+,(SP)+
2187  010104  012700  011554                  MOV     #GGDATO,R0
2188  010110  174010                          STD     AC0,(R0)
2189  010112  012737  011624  001240          MOV     #GGP5,@#STMP3
2190  010120  012737  011624  001242          MOV     #GGP5,@#STMP4
2191  010126  012737  011554  001244          MOV     #GGDATO,@#STMP5
2192  010134  012737  011634  001246          MOV     #GGP6,@#STMP6
2193  010142  104220                  1$:     ERROR   220
2194  010144  000137  011674                  JMP     @#GGDONE
2195
2196  010150                          GGER1:
2197  010150  010537  001252                  MOV     R5,@#STMP10
2198  010154  010437  001254                  MOV     R4,@#STMP11
2199  010160  012737  011624  001240          MOV     #GGP5,@#STMP3
2200  010166  012737  011624  001242          MOV     #GGP5,@#STMP4
2201  010174  012737  011554  001244          MOV     #GGDATO,@#STMP5
2202  010202  012737  011634  001246          MOV     #GGP6,@#STMP6
2203  010210  104207                  1$:     ERROR   207
2204  010212  000137  011674                  JMP     @#GGDONE
2205
2206  010216                          GGER2:
2207  010216  010537  001252                  MOV     R5,@#STMP10
2208  010222  010437  001254                  MOV     R4,@#STMP11
2209  010226  012737  011624  001240          MOV     #GGP5,@#STMP3
2210  010234  012737  011624  001242          MOV     #GGP5,@#STMP4
2211  010242  012737  011554  001244          MOV     #GGDATO,@#STMP5
2212  010250  012737  011634  001246          MOV     #GGP6,@#STMP6
2213  010256  104232                  1$:     ERROR   232
2214  010260  000137  011674                  JMP     @#GGDONE
2215
2216  010264                          GGER3:
2217  010264  010537  001252                  MOV     R5,@#STMP10
2218  010270  010437  001254                  MOV     R4,@#STMP11
2219  010274  012737  011624  001240          MOV     #GGP5,@#STMP3
2220  010302  012737  011624  001242          MOV     #GGP5,@#STMP4
2221  010310  012737  011554  001244          MOV     #GGDATO,@#STMP5
2222  010316  012737  011634  001246          MOV     #GGP6,@#STMP6
2223  010324  104221                  1$:     ERROR   221
2224  010326  000137  011674                  JMP     @#GGDONE
2225
2226  010332  000706          GGER4:  BR      GGER1
2227
2228  010334                          GGER5:
2229  010334  010537  001252                  MOV     R5,@#STMP10
2230  010340  010437  001254                  MOV     R4,@#STMP11
```

```
2231  010344  012737  011624  001240          MOV     #GGP5,@#$TMP3
2232  010352  012737  011624  001242          MOV     #GGP5,@#$TMP4
2233  010360  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
2234  010366  012737  011634  001246          MOV     #GGP6,@#$TMP6
2235  010374  104226                  1$:     ERROR   226
2236  010376  000137  011674                  JMP     @#GGDONE
2237
2238  010402                          GGER6:
2239  010402  010537  001252                  MOV     R5,@#$TMP10
2240  010406  010437  001254                  MOV     R4,@#$TMP11
2241  010412  012737  011624  001240          MOV     #GGP5,@#$TMP3
2242  010420  012737  011624  001242          MOV     #GGP5,@#$TMP4
2243  010426  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
2244  010434  012737  011634  001246          MOV     #GGP6,@#$TMP6
2245  010442  104227                  1$:     ERROR   227
2246  010444  000137  011674                  JMP     @#GGDONE
2247
2248  010450  013701  001236          GGER7:  MOV     @#$TMP2,R1
2249  010454  062701  000002                  ADD     #2,R1
2250  010460  020116                          CMP     R1,(SP)
2251  010462  001402                          BEQ     10$
2252  010464  000137  036614          5$:     JMP     @#FPSPUR
2253  010470                          10$:
2254  010470  170301                          STST    R1
2255  010472  020127  000012                  CMP     R1,#12
2256  010476  001372                          BNE     5$
2257  010500  022626                          CMP     (SP)+,(SP)+
2258  010502  012700  011554                  MOV     #GGDAT0,R0
2259  010506  174010                          STD     AC0,(R0)
2260  010510  012737  011604  001240          MOV     #GGP3,@#$TMP3
2261  010516  012737  011574  001242          MOV     #GGP2,@#$TMP4
2262  010524  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
2263  010532  012737  011634  001246          MOV     #GGP6,@#$TMP6
2264  010540  104224                  1$:     ERROR   224
2265  010542  000137  011674                  JMP     @#GGDONE
2266
2267  010546                          GGER8:
2268  010546  010537  001252                  MOV     R5,@#$TMP10
2269  010552  010437  001254                  MOV     R4,@#$TMP11
2270  010556  012737  011604  001240          MOV     #GGP3,@#$TMP3
2271  010564  012737  011574  001242          MOV     #GGP2,@#$TMP4
2272  010572  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
2273  010600  012737  011634  001246          MOV     #GGP6,@#$TMP6
2274  010606  104207                  1$:     ERROR   207
2275  010610  000137  011674                  JMP     @#GGDONE
2276
2277  010614                          GGER9:
2278  010614  010537  001252                  MOV     R5,@#$TMP10
2279  010620  010437  001254                  MOV     R4,@#$TMP11
2280  010624  012737  011604  001240          MOV     #GGP3,@#$TMP3
2281  010632  012737  011574  001242          MOV     #GGP2,@#$TMP4
2282  010640  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
2283  010646  012737  011634  001246          MOV     #GGP6,@#$TMP6
2284  010654  104232                  1$:     ERROR   232
2285  010656  000137  011674                  JMP     @#GGDONE
2286
```

```
2287  010662                                GGER10:
2288  010662  010537  001252                      MOV     R5,@#$TMP10
2289  010666  010437  001254                      MOV     R4,@#$TMP11
2290  010672  012737  011604  001240              MOV     #GGP3,@#$TMP3
2291  010700  012737  011574  001242              MOV     #GGP2,@#$TMP4
2292  010706  012737  011554  001244              MOV     #GGDAT0,@#$TMP5
2293  010714  012737  011644  001246              MOV     #GGP7,@#$TMP6
2294  010722  104225                        1$:   ERROR   225
2295  010724  000137  011674                      JMP     @#GGDONE
2296
2297  010730                                GGER11:
2298  010730  010537  001252                      MOV     R5,@#$TMP10
2299  010734  010437  001254                      MOV     R4,@#$TMP11
2300  010740  012737  011604  001240              MOV     #GGP3,@#$TMP3
2301  010746  012737  011574  001242              MOV     #GGP2,@#$TMP4
2302  010754  012737  011554  001244              MOV     #GGDAT0,@#$TMP5
2303  010762  012737  011644  001246              MOV     #GGP7,@#$TMP6
2304  010770  104207                        1$:   ERROR   207
2305  010772  000137  011674                      JMP     @#GGDONE
2306
2307  010776                                GGER12:
2308  010776  010537  001252                      MOV     R5,@#$TMP10
2309  011002  010437  001254                      MOV     R4,@#$TMP11
2310  011006  012737  011604  001240              MOV     #GGP3,@#$TMP3
2311  011014  012737  011574  001242              MOV     #GGP2,@#$TMP4
2312  011022  012737  011554  001244              MOV     #GGDAT0,@#$TMP5
2313  011030  012737  011644  001246              MOV     #GGP7,@#$TMP6
2314  011036  104231                        1$:   ERROR   231
2315  011040  000137  011674                      JMP     @#GGDONE
2316
2317  011044                                GGER13:
2318  011044  010537  001252                      MOV     R5,@#$TMP10
2319  011050  010437  001254                      MOV     R4,@#$TMP11
2320  011054  012737  011604  001240              MOV     #GGP3,@#$TMP3
2321  011062  012737  011574  001242              MOV     #GGP2,@#$TMP4
2322  011070  012737  011554  001244              MOV     #GGDAT0,@#$TMP5
2323  011076  012737  011644  001246              MOV     #GGP7,@#$TMP6
2324  011104  104230                        1$:   ERROR   230
2325  011106  000137  011674                      JMP     @#GGDONE
2326
2327  011112  013701  001236              GGER14: MOV     @#$TMP2,R1
2328  011116  062701  000002                      ADD     #2,R1
2329  011122  020116                              CMP     R1,(SP)
2330  011124  001402                              BEQ     10$
2331  011126  000137  036614                5$:   JMP     @#FPSPUR
2332  011132                                10$:
2333  011132  170301                              STST    R1
2334  011134  020127  000012                      CMP     R1,#12
2335  011140  001372                              BNE     5$
2336  011142  022626                              CMP     (SP)+,(SP)+
2337  011144  012700  011554                      MOV     #GGDAT0,R0
2338  011150  174010                              STD     AC0,(R0)
2339
2340  011152  012737  011564  001240              MOV     #GGP1,@#$TMP3
2341  011160  012737  011604  001242              MOV     #GGP3,@#$TMP4
2342  011166  012737  011554  001244              MOV     #GGDAT0,@#$TMP5
```

```
 2343  011174  012737  011634  001246          MOV     #GGP6,@#$TMP6
 2344  011202  104222                  1$:     ERROR   222
 2345  011204  000137  011674                  JMP     @#GGDONE
 2346
 2347  011210                          GGER15:
 2348  011210  010537  001252                  MOV     R5,@#$TMP10
 2349  011214  010437  001254                  MOV     R4,@#$TMP11
 2350  011220  012737  011574  001240          MOV     #GGP2,@#$TMP3
 2351  011226  012737  011654  001242          MOV     #GGP8,@#$TMP4
 2352  011234  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
 2353  011242  012737  011634  001246          MOV     #GGP6,@#$TMP6
 2354  011250  104207                  1$:     ERROR   207
 2355  011252  000137  011674                  JMP     @#GGDONE
 2356
 2357  011256                          GGER16:
 2358  011256  010537  001252                  MOV     R5,@#$TMP10
 2359  011262  010437  001254                  MOV     R4,@#$TMP11
 2360  011266  012737  011574  001240          MOV     #GGP2,@#$TMP3
 2361  011274  012737  011654  001242          MOV     #GGP8,@#$TMP4
 2362  011302  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
 2363  011310  012737  011634  001246          MOV     #GGP6,@#$TMP6
 2364  011316  104232                  1$:     ERROR   232
 2365  011320  000137  011674                  JMP     @#GGDONE
 2366
 2367  011324                          GGER17:
 2368  011324  010537  001252                  MOV     R5,@#$TMP10
 2369  011330  010437  001254                  MOV     R4,@#$TMP11
 2370  011334  012737  011574  001240          MOV     #GGP2,@#$TMP3
 2371  011342  012737  011654  001242          MOV     #GGP8,@#$TMP4
 2372  011350  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
 2373  011356  012737  011664  001246          MOV     #GGP9,@#$TMP6
 2374  011364  104223                  1$:     ERROR   223
 2375  011366  000137  011674                  JMP     @#GGDONE
 2376
 2377  011372                          GGER18:
 2378  011372  010537  001252                  MOV     R5,@#$TMP10
 2379  011376  010437  001254                  MOV     R4,@#$TMP11
 2380  011402  012737  011574  001240          MOV     #GGP2,@#$TMP3
 2381  011410  012737  011654  001242          MOV     #GGP8,@#$TMP4
 2382  011416  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
 2383  011424  012737  011664  001246          MOV     #GGP9,@#$TMP6
 2384  011432  104207                  1$:     ERROR   207
 2385  011434  000137  011674                  JMP     @#GGDONE
 2386
 2387  011440                          GGER19:
 2388  011440  010537  001252                  MOV     R5,@#$TMP10
 2389  011444  010437  001254                  MOV     R4,@#$TMP11
 2390  011450  012737  011574  001240          MOV     #GGP2,@#$TMP3
 2391  011456  012737  011654  001242          MOV     #GGP8,@#$TMP4
 2392  011464  012737  011554  001244          MOV     #GGDAT0,@#$TMP5
 2393  011472  012737  011664  001246          MOV     #GGP9,@#$TMP6
 2394  011500  104230                  1$:     ERROR   230
 2395  011502  000137  011674                  JMP     @#GGDONE
 2396
 2397  011506                          GGER20:
 2398  011506  010537  001252                  MOV     R5,@#$TMP10
```

```
2399  011512  010437  001254                MOV     R4,@#$TMP11
2400  011516  012737  011574  001240        MOV     #GGP2,@#$TMP3
2401  011524  012737  011654  001242        MOV     #GGP8,@#$TMP4
2402  011532  012737  011554  001244        MOV     #GGDAT0,@#$TMP5
2403  011540  012737  011664  001246        MOV     #GGP9,@#$TMP6
2404  011546  104231                1$:     ERROR   231
2405  011550  000137  011674                JMP     @#GGDONE
2406
2407  011554  000000                GGDAT0: 0
2408  011556  000000                        0
2409  011560  000000                        0
2410  011562  000000                        0
2411
2412  011564  000300                GGP1:   300
2413  011566  000000                        0
2414  011570  000000                        0
2415  011572  000000                        0
2416  011574  100200                GGP2:   100200
2417  011576  000000                        0
2418  011600  000000                        0
2419  011602  000000                        0
2420  011604  000200                GGP3:   200
2421  011606  000000                        0
2422  011610  000000                        0
2423  011612  000001                        1
2424  011614  010200                GGP4:   10200
2425  011616  000000                        0
2426  011620  000000                        0
2427  011622  000000                        0
2428  011624  077600                GGP5:   77600           ;OVER FLOW = GGP5 + GGP5
2429  011626  000000                        0
2430  011630  000000                        0
2431  011632  000000                        0
2432  011634  000000                GGP6:   0               ;OVERFLOW RESULT
2433  011636  000000                        0               ;UNDERFLOW RESULT
2434  011640  000000                        0               ;GGP6 = GGP4 + GGP5
2435  011642  000000                        0               ;     = GGP3 + GGP2 (FIU = 0)
2436                                                        ;     = GGP3 + GGP1
2437  011644  062400                GGP7:   62400           ;GGP7 = GGP3 + GGP2 (FIU = 1)
2438  011646  000000                        0
2439  011650  000000                        0
2440  011652  000000                        0
2441  011654  000340                GGP8:   340
2442  011656  000000                        0
2443  011660  000000                        0
2444  011662  000000                        0
2445  011664  000100                GGP9:   100
2446  011666  000000                        0
2447  011670  000000                        0
2448  011672  000000                        0
2449  011674                        GGDONE:
2450  011674  104412                        RSETJP          ;GO INITIALIZE THE FPS AND STACK; AND
2451                                                        ;SEE IF THE USER HAS EXPRESSED
2452                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
2453                                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2454                                                        ;THE USER TYPED CONTROL G?).
```

```
2455
2456
2457                                    ;;*****************************************************************
2458                                    ;*TEST 3        LDCFD AND LDCDF TEST
2459                                    ;*
2460                                    ;*THIS IS A TEST OF LDCFD AND LDCDF.
2461                                    ;*
2462                                    ;;*****************************************************************
2463  011676  000004                    TST3:   SCOPE
2464                                    ;TEST FOR CORRECT AUTO INCREMENT CONSTANT.
2465  011700                            HX1:
2466  011700  104413                           LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
2467  011702  012704  000200                   MOV     #200,R4
2468  011706  170104                           LDFPS   R4
2469  011710  012700  013314                   MOV     #HXP1,R0
2470  011714  172410                           LDD     (R0),AC0
2471  011716  012700  013324                   MOV     #HXP2,R0
2472  011722  012737  011730  001236           MOV     #HX2,@#$TMP2
2473  011730  177420                   HX2:    LDCFD   (R0)+,AC0
2474  011732  020027  013330                   CMP     R0,#HXP2+4          ;IS R0 CORRECT
2475  011736  001402                           BEQ     HX3
2476  011740  000137  012672                   JMP     @#HXER1
2477  011744                            HX3:
2478  011744  170205                           STFPS   R5                  ;GET FPS
2479  011746  012700  013304                   MOV     #HXDAT0,R0
2480  011752  174010                           STD     AC0,(R0)            ;GET AC0
2481  011754  012701  013374                   MOV     #HXP7,R1            ;SEE IF RESULT IS
2482  011760  012702  000004                   MOV     #4,R2               ;CORRECT
2483  011764  022120                   HX4:    CMP     (R1)+,(R0)+
2484  011766  001415                           BEQ     HX7
2485  011770  012701  013324                   MOV     #HXP2,R1            ;DID FD GET
2486  011774  012700  013304                   MOV     #HXDAT0,R0          ;COMPLIMENTED?
2487  012000  012702  000004                   MOV     #4,R2
2488  012004  022120                   HX5:    CMP     (R1)+,(R0)+
2489  012006  001402                           BEQ     HX6
2490  012010  000137  012732                   JMP     @#HXER2
2491  012014  077205                   HX6:    SOB     R2,HX5
2492  012016  000137  012762                   JMP     @#HXER3
2493  012022  077220                   HX7:    SOB     R2,HX4
2494  012024  012704  000200                   MOV     #200,R4             ;FPS CORRECT?
2495  012030  020405                           CMP     R4,R5
2496  012032  001402                           BEQ     HX8
2497  012034  000137  013030                   JMP     @#HXER8
2498                                    ;NOW    TEST    LDCDF
2499  012040                            HX8:
2500  012040  104413                           LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
2501  012042  012704  000200                           MOV #200,R4
2502  012046  170104                           LDFPS R4
2503
2504  012050  012700  013314                   MOV     #HXP1,R0
2505  012054  172410                           LDD     (R0),AC0
2506
2507  012056  012700  013324                   MOV     #HXP2,R0
2508  012062  012737  012072  001236           MOV     #HX9,@#$TMP2
2509
2510  012070  170001                           SETF
```

```
2511
2512  012072  177420                HX9:    LDCDF   (R0)+,AC0       ;TEST INSTRUCTION
2513
2514  012074  020027  013334                CMP     R0,#HXP2+10     ;WAS A GOOD
2515  012100  001402                        BEQ     HX10            ;CONSTANT USED
2516  012102  000137  012712                JMP     @#HXER5         ;TO INCREMENT R0?
2517
2518  012106                        HX10:
2519  012106  170205                        STFPS   R5
2520  012110  012700  013304                MOV     #HXDAT0,R0
2521  012114  170011                        SETD
2522  012116  174010                        STD     AC0,(R0)        ;GET RESULT
2523  012120  012701  013404                MOV     #HXP8,R1
2524  012124  012702  000004                MOV     #4,R2
2525  012130  022120                HX11:   CMP     (R1)+,(R0)+     ;IS IT CORRECT?
2526  012132  001415                        BEQ     HX14
2527
2529  012134  012701  013374                MOV     #HXP7,R1
2529  012140  012700  013304                MOV     #HXDAT0,R0
2530  012144  012702  000004                MOV     #4,R2
2531  012150  022110                HX12:   CMP     (R1)+,(R0)      ;DID FD FAIL TO GET
2532  012152  001402                        BEQ     HX13            ;COMPLIMENTED?
2533  012154  000137  013046                JMP     @#HXER6
2534  012160  077205                HX13:   SOB     R2,HX12
2535  012162  000137  013076                JMP     @#HXER7
2536
2537  012166  077220                HX14:   SOB     R2,HX11
2538
2539  012170  012704  000000                MOV     #0,R4           ;FPS CORRECT?
2540  012174  020405                        CMP     R4,R5
2541  012176  001402                        BEQ     HX15
2542  012200  000137  013030                JMP     @#HXER8
2543
2544                                ;TEST GR7 IMMEDIATE MODE CONSTANT
2545
2546  012204                        HX15:
2547  012204  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2548
2549  012206  012704  000200                MOV     #200,R4
2550  012212  170104                        LDFPS   R4              ;SET FD
2551  012214  012737  012232  001236        MOV     #HX16,@#STMP2
2552  012222  012737  013126  000004        MOV     #HXER9,@#ERRVECT
2553  012230  005001                        CLR     R1
2554  012232  177427  043243        HX16:   LDCFD   #5201,AC0
2555  012236  005201                HX165:  INC     R1
2556  012240  005201                        INC     R1
2557  012242  005201                        INC     R1
2558  012244  012737  036646  000004        MOV     #CPSPUR,@#ERRVECT
2559  012252  020127  000003                CMP     R1,#3           ;SEE IF PC WAS
2560  012256  001402                        BEQ     HX17            ;CORRECT
2561  012260  000137  013162                JMP     @#HXER10
2562
2563  012264                        HX17:
2564  012264  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2565
2566  012266  012704  000200                MOV     #200,R4
```

```
2567  012272  170104                          LDFPS   R4
2568  012274  012737  012322  001236          MOV     #HX18,@#STMP2
2569  012302  012700  013364                  MOV     #HXP6,R0
2570  012306  172410                          LDD     (R0),AC0
2571  012310  012737  036646  000004          MOV     #CPSPUR,@#ERRVECT
2572  012316  012700  013324                  MOV     #HXP2,R0
2573  012322  177410                  HX18:   LDCFD   (R0),AC0
2574
2575  012324  012700  013304                  MOV     #HXDAT0,R0
2576  012330  174010                          STD     AC0,(R0)        ;GET RESULT.
2577  012332  012701  013374                  MOV     #HXP7,R1
2578  012336  012702  000004                  MOV     #4,R2
2579  012342  022021                  HX19:   CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
2580  012344  001402                          BEQ     HX20
2581  012346  000137  012732                  JMP     @#HXER2
2582  012352  077205                  HX20:   SOB     R2,HX19
2583
2584                                   ;TEST LDCFD WITH NEGATIVE OPERAND
2585  012354                           HX21:
2586  012354  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2587  012356  012704  000200                  MOV     #200,R4
2588  012362  170104                          LDFPS   R4
2589  012364  012737  012404  001236          MOV     #HX22,@#STMP2
2590  012372  012700  013364                  MOV     #HXP6,R0
2591  012376  172410                          LDD     (R0),AC0
2592  012400  012700  013344                  MOV     #HXP4,R0
2593  012404  177410                  HX22:   LDCFD   (R0),AC0
2594
2595  012406  012700  013304                  MOV     #HXDAT0,R0
2596  012412  174010                          STD     AC0,(R0)        ;GET RESULT
2597
2598  012414  012701  013354                  MOV     #HXP5,R1
2599  012420  012702  000004                  MOV     #4,R2
2600  012424  022120                  HX23:   CMP     (R1)+,(R0)+
2601  012426  001415                          BEQ     HX26
2602
2603  012430  012701  013374                  MOV     #HXP7,R1
2604  012434  012700  013304                  MOV     #HXDAT0,R0
2605  012440  012702  000004                  MOV     #4,R2
2606  012444  022120                  HX24:   CMP     (R1)+,(R0)+     ;WAS SIGN INCORRECT
2607  012446  001402                          BEQ     HX25
2608  012450  000137  013214                  JMP     @#HXER11
2609  012454  077205                  HX25:   SOB     R2,HX24
2610  012456  000137  013234                  JMP     @#HXER12
2611
2612  012462  077220                  HX26:   SOB     R2,HX23
2613
2614                                   ;TEST   LDCFD   0
2615
2616  012464                           HX27:
2617  012464  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2618  012466  012704  000200                  MOV     #200,R4
2619  012472  170104                          LDFPS   R4
2620
2621  012474  012700  013314                  MOV     #HXP1,R0
2622  012500  172410                          LDD     (R0),AC0
```

```
2623  012502  172010                          ADDD    (R0),AC0
2624
2625  012504  012737  012516  001236          MOV     #HX28,@#STMP2
2626  012512  012700  013314                  MOV     #HXP1,R0
2627  012516  177410                  HX28:   LDCFD   (R0),AC0
2628
2629  012520  170205                          STFPS   R5
2630
2631  012522  012700  013304                  MOV     #HXDAT0,R0
2632  012526  174010                          STD     AC0,(R0)        ;GET RESULT
2633
2634  012530  012701  013314                  MOV     #HXP1,R1
2635  012534  012702  000004                  MOV     #4,R2
2636  012540  022120                  HX29:   CMP     (R1)+,(R0)+     ;IS IT 0?
2637  012542  001402                          BEQ     HX30
2638  012544  000137  013264                  JMP     @#HXER13
2639  012550  077205                  HX30:   SOB     R2,HX29
2640
2641  012552  012704  000204                  MOV     #204,R4         ;FPS CORRECT
2642  012556  020405                          CMP     R4,R5
2643  012560  001402                          BEQ     HX31
2644  012562  000137  013012                  JMP     @#HXER4
2645
2646                                  ;TEST   LDCFD   0
2647
2648  012566                          HX31:
2649  012566  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2650  012570  012704  000200                  MOV     #200,R4
2651  012574  170104                          LDFPS   R4
2652
2653  012576  012700  013364                  MOV     #HXP6,R0
2654  012602  172410                          LDD     (R0),AC0
2655
2656  012604  012737  012616  001236          MOV     #HX32,@#STMP2
2657  012612  012700  013314                  MOV     #HXP1,R0
2658  012616  177410                  HX32:   LDCFD   (R0),AC0
2659
2660  012620  170205                          STFPS   R5
2661
2662  012622  012700  013304                  MOV     #HXDAT0,R0
2663  012626  174010                          STD     AC0,(R0)        ;GET RESULT
2664
2665  012630  012701  013314                  MOV     #HXP1,R1
2666  012634  012702  000004                  MOV     #4,R2
2667  012640  022120                  HX33:   CMP     (R1)+,(R0)+     ;IS IT ZERO?
2668  012642  001402                          BEQ     HX34
2669  012644  000137  013264                  JMP     @#HXER13
2670  012650  077205                  HX34:   SOB     R2,HX33
2671
2672  012652  012704  000204                  MOV     #204,R4         ;FPS CORRECT^
2673  012656  020405                          CMP     R4,R5
2674  012660  001402                          BEQ     HX35
2675  012662  000137  013012                  JMP     @#HXER4
2676  012666  000137  013414          HX35:   JMP     @#HXDONE
2677
2678                                  ;R0 INCORRECT
```

```
2679
2680  012672  012737  013330  001242  HXER1:  MOV     #HXP2+4,@#$TMP4
2681  012700  010037  001240                  MOV     R0,@#$TMP3
2682  012704  104234                  1$:     ERROR   234
2683  012706  000137  013414                  JMP     @#HXDONE
2684
2685  012712  012737  013334  001242  HXER5:  MOV     #HXP2+10,@#$TMP4
2686  012720  010037  001240                  MOV     R0,@#$TMP3
2687  012724  104237                  1$:     ERROR   237
2688  012726  000137  013414                  JMP     @#HXDONE
2689                                ;REPORT BAD DATA
2690  012732  012737  013324  001244  HXER2:  MOV     #HXP2,@#$TMP5
2691  012740  012737  013374  001250          MOV     #HXP7,@#$TMP7
2692  012746  012737  013304  001246  HXER22: MOV     #HXDAT0,@#$TMP6
2693  012754  104233                  1$:     ERROR   233
2694  012756  000137  013414                  JMP     @#HXDONE
2695                                ;
2696  012762  012737  013324  001244  HXER3:  MOV     #HXP2,@#$TMP5
2697  012770  012737  013374  001250          MOV     #HXP7,@#$TMP7
2698  012776  012737  013304  001246  HXER33: MOV     #HXDAT0,@#$TMP6
2699  013004  104241                  1$:     ERROR   241
2700  013006  000137  013414                  JMP     @#HXDONE
2701
2702  013012  010537  001240          HXER4:  MOV     R5,@#$TMP3
2703  013016  010437  001242                  MOV     R4,@#$TMP4
2704  013022  104240                  1$:     ERROR   240
2705  013024  000137  013414                  JMP     @#HXDONE
2706
2707  013030  010537  001240          HXER8:  MOV     R5,@#$TMP3
2708  013034  010437  001242                  MOV     R4,@#$TMP4
2709  013040  104242                  1$:     ERROR   242
2710  013042  000137  013414                  JMP     @#HXDONE
2711  013046  012737  013324  001244  HXER6:  MOV     #HXP2,@#$TMP5
2712  013054  012737  013404  001250          MOV     #HXP8,@#$TMP7
2713  013062  012737  013304  001246  HXER66: MOV     #HXDAT0,@#$TMP6
2714  013070  104244                  1$:     ERROR   244
2715  013072  000137  013414                  JMP     @#HXDONE
2716
2717  013076  012737  013324  001244  HXER7:  MOV     #HXP2,@#$TMP5
2718  013104  012737  013404  001250          MOV     #HXP8,@#$TMP7
2719  013112  012737  013304  001246          MOV     #HXDAT0,@#$TMP6
2720  013120  104243                  1$:     ERROR   243
2721  013122  000137  013414                  JMP     @#HXDONE
2722
2723  013126  032716  000001          HXER9:  BIT     #1,(SP)         ;SEE IF IT
2724  013132  001005                          BNE     1$              ;AN ODD ADDRESS
2725  013134  022716  012236                  CMP     #HX165,(SP)
2726  013140  001402                          BEQ     1$
2727  013142  000137  036646                  JMP     @#CPSPUR
2728
2729  013146  011637  001236          1$:     MOV     (SP),@#$TMP2
2730  013152  022626                  2$:     CMP     (SP)+,(SP)+
2731  013154  104235                  2$:     ERROR   235
2732  013156  000137  013414                  JMP     @#HXDONE
2733
2734  013162  162701  000003          HXER10: SUB     #3,R1
```

```
2735  013166  006301                          ASL     R1
2736  013170  012702  012236                  MOV     #HX165,R2
2737  013174  010237  001242                  MOV     R2,@#STMP4
2738  013200  160102                          SUB     R1,R2
2739  013202  010237  001240                  MOV     R2,@#STMP3
2740  013206  104236              1$:         ERROR   236
2741  013210  000137  013414                  JMP     @#HXDONE
2742
2743  013214  012737  013344  001244  HXER11: MOV     #HXP4,@#STMP5
2744  013222  012737  013354  001250          MOV     #HXP5,@#STMP7
2745  013230  000137  012746                  JMP     @#HXER22
2746  013234  012737  013344  001244  HXER12: MOV     #HXP4,@#STMP5
2747  013242  012737  013354  001250          MOV     #HXP5,@#STMP7
2748  013250  012737  013304  001246          MOV     #HXDAT0,@#STMP6
2749  013256  104245              1$:         ERROR   245
2750  013260  000137  013414                  JMP     @#HXDONE
2751
2752  013264  012737  013314  001244  HXER13: MOV     #HXP1,@#STMP5
2753  013272  012737  013314  001250          MOV     #HXP1,@#STMP7
2754  013300  000137  012746                  JMP     @#HXER22
2755
2756  013304  000000              HXDAT0:     0
2757  013306  000000                          0
2758  013310  000000                          0
2759  013312  000000                          0
2760
2761  013314  000000              HXP1:       0
2762  013316  000000                          0
2763  013320  000000                          0
2764  013322  000000                          0
2765
2766  013324  000577              HXP2:       577
2767  013326  177776                          177776
2768  013330  177777                          177777
2769  013332  177776                          177776
2770  013334  005201              HXP3:       5201
2771  013336  000000                          0
2772  013340  000000                          0
2773  013342  000000                          0
2774  013344  100577              HXP4:       100577
2775  013346  177776                          177776
2776  013350  177777                          177777
2777  013352  177776                          177776
2778  013354  100577              HXP5:       100577
2779  013356  177776                          177776
2780  013360  000000                          0
2781  013362  000000                          0
2782  013364  000252              HXP6:       252
2783  013366  125252                          125252
2784  013370  125252                          125252
2785  013372  125252                          125252
2786
2787  013374  000577              HXP7:       577
2788  013376  177776                          177776
2789  013400  000000                          0
2790  013402  000000                          0
```

```
2791  013404  000577                  HXP8:   577
2792  013406  177777                          177777
2793  013410  000000                          0
2794  013412  000000                          0
2795
2796  013414                          HXDONE:
2797  013414  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
2798                                                                  ;SEE IF THE USER HAS EXPRESSED
2799                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
2800                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2801                                                                  ;THE USER TYPED CONTROL G?).
2802
2803
2804
2805
2806                                    ;;*****************************************************************
2807                                    ;*TEST 4        CMPD TEST
2808                                    ;*
2809                                    ;*THIS IS A TEST OF THE CMPD INSTRUCTION. NOTE THAT A SUBROUTINE
2810                                    ;*IS USED TO SET UP OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE
2811                                    ;*RESULTS
2812                                    ;*
2813                                    ;;*****************************************************************
2814  013416  000004                  TST4:   SCOPE
2815
2816                                    ;TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)
2817  013420                          AAA1:
2818  013420  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2819  013422  004737  014232                  JSR     PC,@#CMPSUB
2820  013426  000000  000000  000000  1$:     .WORD   0,0,0,0                 ;AC0
2821  013434  000000
2822  013436  000000  000000  000000  2$:     .WORD   0,0,0,0                 ;FSRC
2823  013444  000000
2824  013446  000200                  3$:     200                             ;FPS BEFORE EXECUTION
2825  013450  000204                          204                             ;FPS AFTER EXECUTION
2826  013452  000200                          200                             ;ERROR FPS
2827  013454  104001                  4$:     ERROR   1                       ;FPS ERROR
2828
2829
2830                                    ;TEST CMPD WITH (AC=0) AND FSRC POSITIVE.
2831  013456                          AAA2:
2832  013456  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2833  013460  004737  014232                  JSR     PC,@#CMPSUB
2834  013464  000000  000000  000000  1$:     .WORD   0,0,0,0                 ;AC
2835  013472  000000
2836  013474  025252                  2$:     25252                           ;FSRC
2837  013476  052525                          52525
2838  013500  125252                          125252
2839  013502  052525                          52525
2840  013504  000200                  3$:     200                             ;FPS BEFORE EXECUTION
2841  013506  000200                          200                             ;FPS AFTER EXECUTION
2842  013510  000210                          210                             ;ERROR FPS
2843  013512  104003                  4$:     ERROR   3                       ;FPS ERROR
2844
2845                                    ;TEST CMPD WITH (AC=0) AND FSRC NEGATIVE
2846  013514                          AAA3:
```

```
2847  013514  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2848  013516  004737  014232                  JSR     PC,@#CMPSUB
2849  013522  000000  000000  000000  1$:     .WORD   0.0.0.0                 ;AC
2850  013530  000000
2851  013532  125252                  2$:     125252                          ;FSRC
2852  013534  125252                          125252
2853  013536  052525                          52525
2854  013540  125252                          125252
2855  013542  000200                  3$:     200                             ;FPS BEFORE EXECUTION
2856  013544  000210                          210                             ;FPS AFTER EXECUTION
2857  013546  000200                          200                             ;ERROR FPS
2858  013550  104004                  4$:     ERROR   4                       ;FPS ERROR.
2859
2860                                  ;TEST CMPD WITH (FSRC=0) AND AC POSITIVE
2861  013552                          AAA4:
2862  013552  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2863  013554  004737  014232                  JSR     PC,@#CMPSUB
2864  013560  025252                  1$:     25252                           ;AC
2865  013562  052525                          52525
2866  013564  125252                          125252
2867  013566  052525                          52525
2868  013570  000000  000000  000000  2$:     .WORD   0.0.0.0                 ;FSRC
2869  013576  000000
2870  013600  000200                  3$:     200                             ;FPS BEFORE EXECUTION
2871  013602  000210                          210                             ;FPS AFTER EXECUTION
2872  013604  000200                          200                             ;ERROR FPS
2873  013606  104005                  4$:     ERROR   5                       ;FPS ERROR
2874
2875
2876                                  ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
2877  013610                          AAA5:
2878  013610  104413                          LPERR                   ;SET UP THE LCOP ON ERROR ADDRESS.
2879  013612  004737  014232                  JSR     PC,@#CMPSUB
2880  013616  125252                  1$:     125252                          ;AC
2881  013620  125252                          125252
2882  013622  052525                          52525
2883  013624  125252                          125252
2884  013626  000000  000000  000000  2$:     .WORD   0.0.0.0                 ;FSRC
2885  013634  000000
2886  013636  000200                  3$:     200                             ;FPS BEFORE EXECUTION
2887  013640  000200                          200                             ;FPS AFTER EXECUTION
2888  013642  000210                          210                             ;ERROR FPS
2889  013644  104006                  4$:     ERROR   6                       ;FPS ERROR
2890
2891                                  ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
2892  013646                          AAA6:
2893  013646  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2894  013650  004737  014232                  JSR     PC,@#CMPSUB
2895  013654  052525                  1$:     52525                           ;AC
2896  013656  125252                          125252
2897  013660  052525                          52525
2898  013662  125252                          125252
2899  013664  125252                  2$:     125252                          ;:FSRC
2900  013666  052525                          52525
2901  013670  125252                          125252
2902  013672  052525                          52525
```

```
013674  000200          3$:     200                             ;FPS BEFORE EXECUTION
013676  000210                  210                             ;FPS AFTER EXECUTION
013700  000200                  200                             ;ERROR FPS
013702  104007          4$:     ERROR   7                       ;FPS ERROR

                        ;TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
013704                  AAA7:
013704  104413                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
013706  004737  014232          JSR     PC,@#CMPSUB
013712  125252          1$:     125252                          ;AC
013714  052525                  52525
013716  125252                  125252
013720  052525                  52525
013722  052525          2$:     52525                           ;FSRC
013724  125252                  125252
013726  052525                  52525
013730  125252                  125252
013732  000200          3$:     200                             ;FPS BEFORE EXECUTION
013734  000200                  200                             ;FPS AFTER EXECUTION
013736  000210                  210                             ;ERROR FPS
013740  104010          4$:     ERROR   10                      ;FPS ERROR.

                        ;TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
                        ;AND EAC LESS THAN EFSRC.
013742                  AAA8:
013742  104413                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
013744  004737  014232          JSR     PC,@#CMPSUB
013750  012345          1$:     12345                           ;AC
013752  067654                  67654
013754  032101                  32101
013756  023456                  23456
013760  023456          2$:     23456                           ;FSRC
013762  076543                  76543
013764  021012                  21012
013766  034567                  34567
013770  000200          3$:     200                             ;FPS BEFORE EXECUTION
013772  000200                  200                             ;FPS AFTER EXECUTION
013774  000210                  210                             ;ERROR FPS
013776  104011          4$:     ERROR   11                      ;FPS ERROR

                        ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC
014000                  AAA9:
014000  104413                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
014002  004737  014232          JSR     PC,@#CMPSUB
014006  045676          1$:     45676                           ;AC
014010  054321                  54321
014012  012345                  12345
014014  067654                  67654
014016  034567          2$:     34567                           ;FSRC
014020  065432                  65432
014022  101234                  101234
014024  056765                  56765
014026  000200          3$:     200                             ;FPS BEFORE EXECUTION
014030  000210                  210                             ;FPS AFTER EXECUTION
```

```
2959  014032  000200                        200                             ;ERROR FPS
2960  014034  104013                  4$:   ERROR   12
2961
2962                                  ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
2963  014036                          AAA10:
2964  014036  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2965  014040  004737  014232                JSR     PC,@#CMPSUB
2966  014044  012345                  1$:   12345                           ;AC
2967  014046  067012                        67012
2968  014050  034567                        34567
2969  014052  012345                        012345
2970  014054  012345                  2$:   12345                           ;FSRC
2971  014056  067012                        67012
2972  014060  034567                        34567
2973  014062  012345                        012345
2974  014064  000200                  3$:   200                             ;FPS BEFORE EXECUTION
2975  014066  000204                        204                             ;FPS AFTER EXECUTION
2976  014070  000200                        200                             ;ERROR FPS
2977  014072  104013                  4$:   ERROR   13                      ;FPS ERROR
2978
2979                                  ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
2980                                  ;AND FSRC GREATER THAN AC.
2981  014074                          AAA11:
2982  014074  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
2983  014076  004737  014232                JSR     PC,@#CMPSUB
2984  014102  012345                  1$:   12345                           ;AC
2985  014104  067012                        67012
2986  014106  034567                        34567
2987  014110  012345                        012345
2988  014112  012345                  2$:   12345                           ;FSRC
2989  014114  070123                        70123
2990  014116  045670                        45670
2991  014120  123456                        123456
2992  014122  000200                  3$:   200                             ;FPS BEFORE EXECUTION
2993  014124  000200                        200                             ;FPS AFTER EXECUTION
2994  014126  000210                        210                             ;ERROR FPS
2995  014130  104014                  4$:   ERROR   14                      ;FPS ERROR
2996
2997                                  ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
2998                                  ;AND AC GREATER THAN FSRC.
2999  014132                          AAA12:
3000  014132  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3001  014134  004737  014232                JSR     PC,@#CMPSUB
3002  014140  054321                  1$:   54321                           ;AC
3003  014142  076543                        76543
3004  014144  021076                        21076
3005  014146  054321                        54321
3006  014150  054321                  2$:   54321                           ;FSRC
3007  014152  065432                        65432
3008  014154  107654                        107654
3009  014156  032107                        32107
3010  014160  000200                  3$:   200                             ;FPS BEFORE EXECUTION
3011  014162  000210                        210                             ;FPS AFTER EXECUTION
3012  014164  000200                        200                             ;ERROR FPS
3013  014166  104015                  4$:   ERROR   15                      ;FPS ERROR
3014
```

```
                                        ;TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC.
                                        ;AND AC GREATER THAN FSRC
014170                                  AAA13:
014170  104413                                LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
014172  004737  014232                        JSR     PC,@#CMPSUB
014176  112345                        1$:     112345                  ;AC
014200  043210                                43210
014202  076543                                76543
014204  021076                                21076
014206  112345                        2$:     112345                  ;FSRC
014210  054321                                54321
014212  007654                                07654
014214  032107                                32107
014216  000200                        3$:     200                     ;FPS BEFORE EXECUTION
014220  000210                                210                     ;FPS AFTER EXECUTION
014222  000200                                200                     ;ERROR FPS
014224  104016                        4$:     ERROR   16              ;FPS ERROR

014226  000137  014422                        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.

014232  012601                        CMPSUB: MOV     (SP)+,R1        ;PICK UP A POINTER TO THE
                                                                        ;ARGUMENTS.
014234  016100  000020                        MOV     20(R1),R0       ;GET THE FPS BEFORE EXECUTION.
014240  170100                                LDFPS   R0              ;LOAD IT INTO THE FPS.

014242  012737  014264  001236                MOV     #1$,@#STMP2     ;SAVE ADDRESS OF CMPD INSTRUCTION.
014250  010100                                MOV     R1,R0           ;GET ADDRESS OF AC OPERAND.
014252  172410                                LDD     (R0),AC0        ;LOAD AC0 OPERAND

014254  010100                                MOV     R1,R0           ;COMPUTE FSRC OPERAND
014256  062700  000010                        ADD     #10,R0          ;ADDRESS
```

```
3071
3072  014262  000240                          NOP                     ;FOR SCOPING.
3073  014264  173410                  1$:     CMPD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
3074
3075  014266  170205                          STFPS   R5              ;SAVE FPS AFTER INSTRUCTION.
3076
3077  014270  016104  000022                  MOV     22(R1),R4       ;GET EXPECTED FPS.
3078                                                                  ;IF INCORRECT SET UP FOR
3079  014274  010137  001240                  MOV     R1,@#$TMP3      ;AN ERROR CALL.
3080  014300  010137  001242                  MOV     R1,@#$TMP4
3081  014304  062737  000010  001242          ADD     #10,@#$TMP4
3082  014312  010537  001244                  MOV     R5,@#$TMP5
3083  014316  010437  001246                  MOV     R4,@#$TMP6
3084  014322  020405                          CMP     R4,R5           ;WAS FPS CORRECT?
3085  014324  001410                          BEQ     3$              ;BRANCH IF YES.
3086
3087
3088  014326  026105  000024                  CMP     24(R1),R5       ;WAS THE FPS THE SAME
3089                                                                  ;AS THE EXPECTED INCORRECT FPS?
3090  014332  001003                          BNE     2$              ;BRANCH IF NO MATCH.
3091
3092  014334  062701  000026                  ADD     #26,R1          ;IF THE EXPECTED INCORRECT
3093                                                          ;FPS MATCHED THE RESULTANT FPS
3094  014340  000111                          JMP     (R1)            ;RETURN TO THE ERROR CALL
3095                                                                  ;IN THE CALLING ROUTINE.
3096
3097  014342  104001                  2$:     ERROR   1               ;OTHERWISE REPORT INCORRECT FPS
3098  014344  000411                          BR      5$
3099
3100  014346  012700  014412          3$:     MOV     #CMPTMP,R0      ;IF FPS WAS CORRECT MAKE SURE
3101  014352  174010                          STD     AC0,(R0)        ;AC0 WAS NOT AFFECTED BY CMPD.
3102  014354  010102                          MOV     R1,R2
3103  014356  012703  000004                  MOV     #4,R3
3104  014362  022220                  4$:     CMP     (R2)+,(R0)+
3105  014364  001003                          BNE     6$
3106  014366  077303                          SOB     R3,4$
3107
3108  014370  000161  000030          5$:     JMP     30(R1)          ;RETURN
3109
3110  014374                          6$:                             ;REPORT AC0 MODIFIED BY CMPD
3111  014374  010137  001240                  MOV     R1,@#$TMP3
3112  014400  012737  014412  001242          MOV     #CMPTMP,@#$TMP4
3113  014406  104002                  7$:     ERROR   2
3114  014410  000767                          BR      5$              ;RETURN
3115
3116  014412  000000  000000  000000  CMPTMP: .WORD   0,0,0,0
3117  014420  000000
3118
3119
3120
3121  014422                          AAADONE:
3122  014422  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
3123                                                                  ;SEE IF THE USER HAS EXPRESSED
3124                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
3125                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3126                                                                  ;THE USER TYPED CONTROL G?).
```

```
                                        ;;*****************************************************************
                                        ;*TEST 5        DIVD WITH (FSRC=0) AND (BUT FC) TEST
                                        ;*
                                        ;*THIS IS A TEST OF THE DIVD INSTRUCTION WITH A
                                        ;*ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH
                                        ;*TRAP ENABLED AND TRAPS DISABLED.
                                        ;*
                                        ;;*****************************************************************
014424  000004                TSTS:   SCOPE

                                ;FIRST TEST DIVD WITH (FSRC=AC=0) AND TRAPS DISABLED.
014426  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
014430  012704  040200          BBB0:   MOV     #40200,R4       ;SET UP FPS
                                                                ;WITH INTERRUPTS
                                                                ;DISABLED.
014434  170104                          LDFPS   R4
014436  012737  014700  000244          MOV     #BBBER1,@#FPVECT;SET UP FOR ANY FP INTERRUPTS.
014444  012737  014464  001236          MOV     #BBB1,@#$TMP2
014452  012700  015104                  MOV     #BBBP1,R0       ;SET UP AC0 = 0
014456  172410                          LDD     (R0),AC0
014460  012701  015104                  MOV     #BBBP1,R1       ;FSRC = 0

014464  174411                  BBB1:   DIVD    (R1),AC0        ;TEST INSTRUCTION

014466  170205                          STFPS   R5              ;GET FPS
014470  170303                          STST    R3              ;GET FEC

014472  012704  140204                  MOV     #140204,R4      ;EXPECTED FPS.
014476  020405                          CMP     R4,R5           ;IS FPS CORRECT.
014500  001131                          BNE     BBBER2          ;IF INCORRECT BRANCH.

014502  012702  000004                  MOV     #4,R2           ;EXPECTED FEC.
014506  020203                          CMP     R2,R3           ;IS FEC CORRECT?
014510  001140                          BNE     BBBER3          ;IF INCORRECT BRANCH.

                                ;TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
014512                          BBB2:
014512  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
014514  012704  040200                  MOV     #40200,R4       ;LOAD FPS WITH TRAPS DISABLED.
014520  170104                          LDFPS   R4

014522  012737  014542  001236          MOV     #BBB3,@#$TMP2
014530  012700  015114                  MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO).
014534  172410                          LDD     (R0),AC0
014536  012700  015104                  MOV     #BBBP1,R0       ;FSRC=0
014542  174410                  BBB3:   DIVD    (R0),AC0

014544  170205                          STFPS   R5              ;GET FPS.
014546  170303                          STST    R3              ;GET FEC.

014550  012704  140200                  MOV     #140200,R4              ;EXPECTED FPS.
014554  020405                          CMP     R4,R5           ;IS FPS CORRECT?
```

```
3183  014556  001102                        BNE     BBBER2          ;IF INCORRECT BRANCH.
3184
3185  014560  012702  000004                MOV     #4,R2           ;EXPECTED FEC.
3186  014564  020203                        CMP     R2,R3           ;WAS FEC CORRECT?
3187  014566  001111                        BNE     BBBER3          ;IF INCORRECT BRANCH.
3188
3189                                ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
3190  014570                        BBB4:
3191  014570  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3192  014572  012704  000200                MOV     #200,R4         ;SET UP FPS. TRAP ENABLED.
3193  014576  170104                        LDFPS   R4
3194
3195  014600  012737  014626  001236        MOV     #BBB5,@#$TMP2
3196  014606  012700  015114                MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO..
3197  014612  172410                        LDD     (R0),AC0
3198
3199  014614  012737  014634  000244        MOV     #BBB6,@#FPVECT  ;SET UP FOR THE EXPECTED INTERRUPT.
3200  014622  012700  015104                MOV     #BBBP1,R0       ;FSRC=0
3201
3202  014626  174410                BBB5:   DIVD    (R0),AC0        ;TEST INSTRUCTION (SHOULD RESULT IN TRAP..
3203  014630  170000                        CFCC
3204
3205  014632  000502                        BR      BBBER4          ;GO REPORT FAILURE, NO TRAP.
3206
3207  014634  022716  014630        BBB6:   CMP     #BBB5+2,(SP)    ;TRAP TO HERE WHEN THE DIVISION BY 0
3208                                                                ;OCCURS. FIRST SEE IF THE ADDRESS OF
3209                                                                ;THE TRAP IS 2+THE ADDRESS OF THE TEST
3210                                                                ;DIVD INSTRUCTION.
3211  014640  001402                        BEQ     1$
3212  014642  000137  036614                JMP     @#FPSPUR        ;IF NOT THEN REPORT AN UNEXPECTED
3213                                                                ;FP TRAP.
3214  014646  170205                1$:     STFPS   R5              ;GET FPS.
3215  014650  170303                        STST    R3              ;GET FEC.
3216  014652  022626                        CMP     (SP)+,(SP)+     ;RESET THE STACK.
3217
3218  014654  012704  100200                MOV     #100200,R4      ;EXPECTED FPS.
3219  014660  020405                        CMP     R4,R5           ;IS FPS CORRECT?
3220  014662  001040                        BNE     BBBER2          ;IF INCORRECT BRANCH.
3221
3222  014664  012702  000004                MOV     #4,R2           ;EXPECTED FEC.
3223  014670  020203                        CMP     R2,R3           ;IS FEC CORRECT?
3224  014672  001047                        BNE     BBBER3          ;IF INCORRECT BRANCH.
3225
3226  014674  000137  015124                JMP     @#BBBDONE       ;OTHERWISE GO TO NEXT TEST.
3227
3228
3229                                ;TRAP HERE IF AN UNEXPECTED INTERRUPT OCCURS.
3230  014700  062737  000002  001236  BBBER1: ADD     #2,@#$TMP2      ;SEE IF THE INTERRUPT OCCURRED
3231                                                                ;DURING THE EXECUTION OF THE DIVD
3232                                                                ;INSTRUCTION BEING TESTED.
3233  014706  021637  001236                CMP     (SP),@#$TMP2
3234  014712  001402                        BEQ     1$
3235  014714  000137  036614                JMP     @#FPSPUR        ;IF NOT REPORT UNEXPECTED FP TRAP.
3236
3237  014720  022626                1$:     CMP     (SP)+,(SP)+     ;RESET THE STACK.
3238  014722  170303                        STST    R3              ;GET FEC.
```

```
3239  014724  170205                          STFPS   R5                      ;GET FPS.
3240  014726  012737  000004  001240          MOV     #4,@#$TMP3              ;EXPECTED FEC.
3241  014734  010337  001242                  MOV     R3,@#$TMP4
3242  014740  010537  001244                  MOV     R5,@#$TMP5
3243  014744  010037  001250                  MOV     R0,@#$TMP7
3244  014750  012737  140200  001246          MOV     #140200,@#$TMP6
3245  014756  104017                  2$:     ERROR   17                      ;REPORT (BUT FD, FAILED RESULTING IN AN FP TRAP
3246                                                                          ;WITH TRAPS DISABLED.
3247  014760  000137  015124                  JMP     @#BBBDONE
3248
3249                                  ;REPORT FPS INCORRECT:
3250  014764  010537  001242          BBBER2: MOV     R5,@#$TMP4
3251  014770  010437  001244                  MOV     R4,@#$TMP5
3252  014774  010037  001246                  MOV     R0,@#$TMP6
3253  015000  010137  001250                  MOV     R1,@#$TMP7
3254  015004  104020                  1$:     ERROR   20
3255  015006  000137  015124                  JMP     @#BBBDONE
3256
3257                                  ;REPORT FEC INCORRECT:
3258  015012  010337  001242          BBBER3: MOV     R3,@#$TMP4
3259  015016  010237  001240                  MOV     R2,@#$TMP3
3260  015022  010037  001246                  MOV     R0,@#$TMP6
3261  015026  010137  001250                  MOV     R1,@#$TMP7
3262  015032  104021                  1$:     ERROR   21
3263  015034  000137  015124                  JMP     @#BBBDONE
3264
3265                                  ;REPORT NO TRAP OCCURRED AFTER TRYING TO DIVIDE
3266                                  ;BY ZERO WITH ALL TRAPS ENABLED.
3267  015040  170303                  BBBER4: STST    R3                      ;GET FEC.
3268  015042  170205                          STFPS   R5                      ;GET FPS.
3269  015044  012737  000004  001242          MOV     #4,@#$TMP4
3270  015052  010337  001240                  MOV     R3,@#$TMP3
3271  015056  010537  001244                  MOV     R5,@#$TMP5
3272  015062  012737  100200  001246          MOV     #100200,@#$TMP6
3273  015070  010037  001250                  MOV     R0,@#$TMP7
3274  015074  010137  001252                  MOV     R1,@#$TMP10
3275  015100  104022                  1$:     ERROR   22
3276  015102  000410                          BR      BBBDONE
3277
3278  015104  000000  000000  000000  BBBP1:  .WORD   0,0,0,0
3279  015112  000000
3280  015114  012345  054321  023456  BBBP2:  .WORD   12345,54321,23456,76543
3281  015122  076543
3282
3283
3284
3285  015124                          BBBDONE:
3286  015124  104412                          RSETUP                          ;GO INITIALIZE THE FPS AND STACK; AND
3287                                                                          ;SEE IF THE USER HAS EXPRESSED
3288                                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
3289                                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3290                                                                          ;THE USER TYPED CONTROL G?).
3291
3292
3293
3294                                  ;;**********************************************************************
```

```
                                        ;*TEST 6        DIVF TEST
                                        ;*
                                        ;*THIS IS A TEST OF THE DIVF INSTRUCTION. NOTE THAT A SUBROUTINE IS
                                        ;*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE
                                        ;*RESULTS.
                                        ;*
                                        ;;*****************************************************************
015126  000004                        TST6:   SCOPE

                                        ;CHECK DIVF WITH (AC=0).
015130                                  CCC1:
015130  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
015132  004767  000552                          JSR     PC,DIVFSUB
015136  000000  000000                  1$:     .WORD   0,0             ;AC
015142  012345  067012                  2$:     .WORD   12345,67012     ;FSRC
015146  000000  000000                  3$:     .WORD   0,0             ;RES
015152  000000                          4$:     0                       ;FPS BEFORE EXECUTION.
015154  000004                                  4                       ;FPS AFTER EXECUTION
015156  012345  067012                  5$:     .WORD   12345,67012     ;ERROR RESULT
015162  104023                          6$:     ERROR   23              ;RESULT BAD.

                                        ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.
015164                                  CCC2:
015164  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
015166  004737  015710                          JSR     PC,@#DIVFSUB
015172  065652  125252                  1$:     .WORD   65652,125252    ;AC
015176  065600  000000                  2$:     .WORD   65600,0         ;FSRC
015202  040252  125252                  3$:     .WORD   40252,125252    ;RES
015206  003000                          4$:     3000                    ;FPS BEFORE EXECUTION.
015210  003000                                  3000                    ;FPS AFTER EXECUTION.
015212  040052  125252                  5$:     .WORD   40052,125252    ;ERROR RESULT.
015216  104024                          6$:     ERROR   24              ;DIV NORMALIZE FAILURE.

                                        ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.
015220                                  CCC3:
015220  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
015222  004767  000462                          JSR     PC,DIVFSUB
015226  076400  000000                  1$:     .WORD   76400,0         ;AC
015232  076400  000000                  2$:     .WORD   76400,0         ;FSRC
015236  040200  000000                  3$:     .WORD   40200,0         ;RES
015242  001000                          4$:     1000                    ;FPS BEFORE EXECUTION.
015244  001000                                  1000                    ;FPS AFTER EXECUTION.
015246  140200  000000                  5$:     .WORD   140200,0        ;ERROR RES.

015252  104025                          6$:     ERROR   25              ;SIGN BAD.

                                        ;TEST DIVF WITH BOTH OPERANDS POSITIVE.
015254                                  CCC4:
015254  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
015256  004737  015710                          JSR     PC,@#DIVFSUB
015262  056777  177777                  1$:     .WORD   56777,177777    ;AC
015266  054200  000000                  2$:     .WORD   54200,0         ;FSRC
015272  042777  177777                  3$:     .WORD   42777,177777    ;RES
015276  000000                          4$:     0                       ;FPS BEFORE EXECUTION.
015300  000000                                  0                       ;FPS AFTER EXECUTION.
015302  002000  002000                  5$:     .WORD   2000,2000       ;ERROR RES.
```

```
3351  015306  104023                  6$:     ERROR   23
3352
3353                                  ;TEST THE DIVF INSTRUCTION:
3354  015310                          CCCE:
3355  015310  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3356  015312  004737  015710                  JSR     PC,@#DIVFSUB
3357  015316  012377  177777          1$:     .WORD   12377,177777    ;AC
3358  015322  012300  000000          2$:     .WORD   12300,0         ;FSRC
3359  015326  040252  125252          3$:     .WORD   40252,125252    ;RES
3360  015332  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
3361  015334  000000                          0                       ;FPS AFTER EXECUTION.
3362  015336  177777  177777          5$:     .WORD   -1,-1           ;ERROR RES.
3363  015342  104023                  6$:     ERROR   23
3364
3365                                  ;TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
3366  015344                          CCC6:
3367  015344  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3368  015346  004737  015710                  JSR     PC,@#DIVFSUB
3369  015352  064600  000001          1$:     .WORD   64600,1         ;AC
3370  015356  066600  000000          2$:     .WORD   66600,0         ;FSRC
3371  015362  036200  000001          3$:     .WORD   36200,1         ;RES
3372  015366  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
3373  015370  000000                          0                       ;FPS AFTER EXECUTION.
3374  015372  003000  003000          5$:     .WORD   3000,3000       ;ERROR RES.
3375  015376  104023                  6$:     ERROR   23
3376
3377                                  ;TEST DIVF.
3378  015400                          CCC7:
3379  015400  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3380  015402  004737  015710                  JSR     PC,@#DIVFSUB
3381  015406  034577  177776          1$:     .WORD   34577,177776    ;AC
3382  015412  023400  000000          2$:     .WORD   23400,0         ;FSRC
3383  015416  051377  177776          3$:     .WORD   51377,177776    ;RES
3384  015422  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
3385  015424  000000                          0                       ;FPS AFTER EXECUTION.
3386  015426  003400  003400          5$:     .WORD   3400,3400       ;ERROR RES.
3387  015432  104023                  6$:     ERROR   23
3388
3389
3390                                  ;DIVF TEST.
3391  015434                          CCC8:
3392  015434  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS
3393  015436  004737  015710                  JSR     PC,@#DIVFSUB
3394  015442  067652  125252          1$:     .WORD   67652,125252    ;AC
3395  015446  056500  000000          2$:     .WORD   56500,0         ;FSRC
3396  015452  051343  107070          3$:     .WORD   51343,107070    ;RES
3397  015456  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
3398  015460  000000                          0                       ;FPS AFTER EXECUTION.
3399  015462  051543  107070          5$:     .WORD   51543,107070    ;ERROR RES.
3400  015466  104026                  6$:     ERROR   26              ;DIDN'T INCREMENT THE EXPONENT
3401                                                                  ;AFTER DIVID NORMALIZATION.
3402
3403                                  ;DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
3404  015470                          CCC9:
3405  015470  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3406  015472  004737  015710                  JSR     PC,@#DIVFSUB
```

```
3407  015476  140400  000000          1$:     .WORD   140400,0        ;AC
3408  015502  140500  000000          2$:     .WORD   140500,0        ;FSRC
3409  015506  040052  125253          3$:     .WORD   040052,125253   ;RES
3410  015512  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
3411  015514  000000                          0                       ;FPS AFTER EXECUTION.
3412  015516  140052  125253          5$:     .WORD   140052,125253   ;ERROR RES.
3413  015522  104027                  6$:     ERROR   27              ;BAD SIGN.
3414
3415                                  ;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
3416  015524                          CCC10:
3417  015524  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3418  015526  004737  015710                  JSR     PC,@#DIVFSUB
3419  015532  160077  000000          1$:     .WORD   160077,0        ;AC
3420  015536  040277  000000          2$:     .WORD   40277,0         ;FSRC
3421  015542  160000  000000          3$:     .WORD   160000,0        ;RES
3422  015546  000007                  4$:     7                       ;FPS BEFORE EXECUTION.
3423  015550  000010                          10                      ;FPS AFTER EXECUTION.
3424  015552  060000  000000          5$:     .WORD   60000,0         ;ERROR RES.
3425  015556  104027                  6$:     ERROR   27              ;BAD SIGN.
3426
3427                                  ;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
3428  015560                          CCC11:
3429  015560  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3430  015562  004737  015710                  JSR     PC,@#DIVFSUB
3431  015566  040400  000000          1$:     .WORD   40400,0         ;AC
3432  015572  140500  000000          2$:     .WORD   140500,0        ;FSRC
3433  015576  140052  125253          3$:     .WORD   140052,125253   ;RES
3434  015602  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
3435  015604  000010                          10                      ;FPS AFTER EXECUTION.
3436  015606  040052  125253          5$:     .WORD   40052,125253    ;ERROR RES.
3437  015612  104027                  6$:     ERROR   27              ;BAD SIGN.
3438
3439
3440                                  ;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
3441  015614                          CCC12:
3442  015614  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3443  015616  004737  015710                  JSR     PC,@#DIVFSUB
3444  015622  060100  000001          1$:     .WORD   60100,1         ;AC
3445  015626  040300  000000          2$:     .WORD   40300,0         ;FSRC
3446  015632  060000  000000          3$:     .WORD   60000,0         ;RES
3447  015636  000052                  4$:     52                      ;FPS BEFORE EXECUTION.
3448  015640  000040                          40                      ;FPS AFTER EXECUTION.
3449  015642  060000  000001          5$:     .WORD   60000,1         ;ERROR RES.
3450  015646  104030                  6$:     ERROR   30              ;TRUNCATION ERROR
3451
3452                                  ;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.
3453  015650                          CCC13:
3454  015650  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3455  015652  004767  000032                  JSR     PC,DIVFSUB
3456  015656  060100  000001          1$:     .WORD   60100,1         ;AC
3457  015662  040300  000000          2$:     .WORD   40300,0         ;FSRC
3458  015666  060000  000001          3$:     .WORD   60000,1         ;RES
3459  015672  000005                  4$:     5                       ;FPS BEFORE EXECUTION.
3460  015674  000000                          0                       ;FPS AFTER EXECUTION.
3461  015676  060000  000000          5$:     .WORD   60000,0         ;ERROR RES.
3462  015702  104031                  6$:     ERROR   31              ;ROUND ERROR.
```

```
3463
3464  015704  000137  016134                    JMP     @#CCCDONE       ;GO TO NEXT TEST.
3465
3466                                  ;THIS SUBROUTINE, DIVFSUB, IS CALLED TO SET UP, EXECUTE
3467                                  ;AND CHECK THE RESULT OF A DIVF INSTRUCTION. IT IS CALLED THUS:
3468                                  ;
3469                                  ;               JSR     PC,@#DIVFSUB
3470                                  ;       ACARG:  .WORD   X,X                     ;AC OPERAND
3471                                  ;       FSRCARG:.WORD   X,X                     ;FSRC OPERAND
3472                                  ;       RES:    .WORD   X,X                     ;EXPECTED RESULT
3473                                  ;       FPSB:   .WORD   X                       ;FPS BEFORE EXECUTION
3474                                  ;       FPSA:   .WORD   X                       ;FPS AFTER EXECUTION
3475                                  ;       ERRES:  .WORD   X,X                     ;ERROR RESULT
3476                                  ;       ERR:    ERROR   X                       ;RESULT ERROR
3477                                  ;       CONT:                                   ;RETURN ADDRESS
3478                                  ;
3479                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
3480                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVF IS EXECUTED.
3481                                  ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
3482                                  ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
3483                                  ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
3484                                  ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
3485                                  ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
3486                                  ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
3487                                  ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
3488                                  ;THEN THE FAILURE IS REPORTED IN DIVFSUB AND CONTROL IS PASSED TO
3489                                  ;CONT. IF NO ERRORS ARE DETECTED THEN DIVFSUB RETURNS CONTROL
3490                                  ;TO CONT.
3491
3492  015710  012601                  DIVFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
3493  015712  012700  000200                  MOV     #200,R0         ;SET FD MODE.
3494  015716  170100                  LDFPS   R0
3495  015720  010100                  MOV     R1,R0           ;LOAD THE AC OPERAND.
3496  015722  172410                  LDD     (R0),AC0
3497  015724  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
3498  015730  170100                  LDFPS   R0
3499  015732  012737  015746  001236        MOV     #1$,@#$TMP2
3500  015740  010100                  MOV     R1,R0
3501  015742  062700  000004                  ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.
3502
3503  015746  174410                  1$:     DIVF    (R0),AC0        ;TEST INSTRUCTION.
3504
3505  015750  170204                  STFPS   R4              ;GET THE FPS.
3506  015752  012700  000200                  MOV     #200,R0         ;SET FD MODE
3507  015756  170100                  LDFPS   R0
3508
3509  015760  012700  016124                  MOV     #DIVFT,R0       ;GET THE RESULT OF THE DIVF.
3510  015764  174010                  STD     AC0,(R0)
3511
3512  015766  010102                  MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
3513  015770  010237  001240                  MOV     R2,@#$TMP3
3514  015774  062702  000004                  ADD     #4,R2
3515  016000  010237  001242                  MOV     R2,@#$TMP4
3516  016004  062702  000004                  ADD     #4,R2
3517  016010  010237  001244                  MOV     R2,@#$TMP5
3518  016014  012737  016124  001246        MOV     #DIVFT,@#$TMP6
```

```
3519  016022  010437  001250                  MOV     R4,@#$TMP7
3520  016026  016137  000016  001252          MOV     16(R1),@#$TMP10
3521
3522  016034  021061  000010                  CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
3523  016040  001011                          BNE     10$             ;IF INCORRECT BRANCH.
3524  016042  026061  000002  000012          CMP     2(R0),12(R1)
3525  016050  001005                          BNE     10$
3526
3527  016052  026104  000016                  CMP     16(R1),R4       ;IS FPS CORRECT?
3528  016056  001020                          BNE     15$             ;IF INCORRECT BRANCH.
3529  016060  000161  000026                  JMP     26(R1)          ;IF NO ERRORS OCCURRED RETURN.
3530
3531  016064  021061  000020          10$:    CMP     (R0),20(R1)     ;DOES THE INCORRECT RESULT
3532  016070  001010                          BNE     11$             ;MATCH THE ANTICIPATED INCORRECT RESULT.
3533  016072  026061  000002  000022          CMP     2(R0),22(R1)
3534  016100  001004                          BNE     11$             ;BRANCH IF NO.
3535
3536  016102  010102                          MOV     R1,R2           ;IT MATCHED SO RETURN TO THE ERROR
3537                                                                  ;REPORT AT THE CALLING ROUTINE.
3538  016104  062702  000024                  ADD     #24,R2
3539  016110  000112                          JMP     (R2)
3540
3541  016112                          11$:                            ;REPORT RESULT INCORRECT.
3542  016112  104023                  12$:    ERROR   23
3543  016114  000161  000026          13$:    JMP     26(R1)
3544
3545  016120                          15$:                            ;REPORT FPS INCORRECT.
3546  016120  104032                  16$:    ERROR   32
3547  016122  000774                          BR      13$
3548
3549  016124  000000  000000  000000  DIVFT:  .WORD   0,0,0,0
3550  016132  000000
3551
3552  016134                          CCCDONE:
3553  016134  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
3554                                                                  ;SEE IF THE USER HAS EXPRESSED
3555                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
3556                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3557                                                                  ;THE USER TYPED CONTROL G?).
3558
3559
3560
3561                                  ;;*****************************************************************
3562                                  ;*TEST 7        DIVD TEST
3563                                  ;*
3564                                  ;*THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS
3565                                  ;*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.
3566                                  ;*
3567                                  ;;*****************************************************************
3568  016136  000004                  TST7:   SCOPE
3569
3570                                  ;DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.
3571  016140                          DDD1:
3572  016140  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3573  016142  004737  016630                  JSR     PC,@#DIVDSUB
3574  016146  034277  000000  000000  1$:     .WORD   34277,0,0,0     ;AC
```

```
016154 000000
016156 040277 000000 000000 2$:    .WORD   40277,0,0,0     ;FSRC
016164 000000
016166 034200 000000 000000 3$:    .WORD   34200,0,0,0     ;RES
016174 000000
016176 000200                      4$:     200             ;FPS BEFORE EXECUTION.
016200 000200                              200             ;FPS AFTER EXECUTION.
016202 177777 177777 177777 5$:    .WORD   -1,-1,-1,-1     ;ERROR RES.
016210 177777
016212 104033                      6$:     ERROR   33

                                   ;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.
016214                             DDD2:
016214 104413                              LPERR           ;SET UP THE LOOP ON ERROR ADDRESS.
016216 004737 016630                       JSR     PC,@#DIVDSUB
016222 134277 000000 000000 1$:    .WORD   134277,0,0,0    ;AC
016230 000000
016232 040277 000000 000000 2$:    .WORD   40277,0,0,0     ;FSRC
016240 000000
016242 134200 000000 000000 3$:    .WORD   134200,0,0,0    ;RES
016250 000000
016252 000207                      4$:     207             ;FPS BEFORE EXECUTION.
016254 000210                              210             ;FPS AFTER EXECUTION.
016256 177777 177777 177777 5$:    .WORD   -1,-1,-1,-1     ;ERROR RESULT
016264 177777
016266 104033                      6$:     ERROR   33

                                   ;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
016270                             DDD3:
016270 104413                              LPERR           ;SET UP THE LOOP ON ERROR ADDRESS.
016272 004767 000332                       JSR     PC,DIVDSUB
016276 134300 000000 000000 1$:    .WORD   134300,0,0,1    ;AC
016304 000001
016306 140300 000000 000000 2$:    .WORD   140300,0,0,0    ;FSRC
016314 000000
016316 034200 000000 000000 3$:    .WORD   34200,0,0,0     ;RES
016324 000000
016326 000250                      4$:     250             ;FPS BEFORE EXECUTION.
016330 000240                              240             ;FPS AFTER EXECUTION.
016332 034200 000000 000000 5$:    .WORD   34200,0,0,1     ;ERROR RES.
016340 000001
016342 104035                      6$:     ERROR   35      ;TRUNCATION ERROR.

                                   ;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
016344                             DDD4:
016344 104413                              LPERR           ;SET UP THE LOOP ON ERROR ADDRESS.
016346 004737 016630                       JSR     PC,@#DIVDSUB
016352 034300 000000 000000 1$:    .WORD   34300,0,0,1     ;AC
016360 000001
016362 140300 000000 000000 2$:    .WORD   140300,0,0,0    ;FSRC
016370 000000
016372 134200 000000 000000 3$:    .WORD   134200,0,0,1    ;RES
016400 000001
016402 000207                      4$:     207             ;FPS BEFORE EXECUTION.
016404 000210                              210             ;FPS AFTER EXECUTION.
016406 134200 000000 000000 5$:    .WORD   134200,0,0,0    ;ERROR RES.
```

```
3631  016414  000000
3632  016416  104036                  6$:     ERROR   36              ;ROUND ERROR.
3633
3634                                  ;DIVD TEST.
3635  016420                          DDD5:
3636  016420  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3637  016422  004737  016630                  JSR     PC,@#DIVDSUB
3638  016426  100400  000000  000000  1$:     .WORD   100400,0,0,0    ;AC
3639  016434  000000
3640  016436  000500  000000  000000  2$:     .WORD   500,0,0,0       ;FSRC
3641  016444  000000
3642  016446  140052  125252          3$:     .WORD   140052,125252   ;RES
3643  016452  125252  125252                  .WORD   125252,125252
3644  016456  007647                  4$:     7647                    ;FPS BEFORE EXECUTION.
3645  016460  007650                          7650                    ;FPS AFTER EXECUTION.
3646  016462  177777  177777  177777  5$:     .WORD   -1,-1,-1,-1     ;ERROR RES.
3647  016470  177777
3648  016472  104033                  6$:     ERROR   33
3649
3650
3651                                  ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
3652  016474                          DDD6:
3653  016474  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3654  016476  004737  016630                  JSR     PC,@#DIVDSUB
3655  016502  000400  000000  000000  1$:     .WORD   400,0,0,0       ;AC
3656  016510  000000
3657  016512  100500  000000  000000  2$:     .WORD   100500,0,0,0    ;FSRC
3658  016520  000000
3659  016522  140052  125252          3$:     .WORD   140052,125252   ;RES
3660  016526  125252  125253                  .WORD   125252,125253
3661  016532  007707                  4$:     7707                    ;FPS BEFORE EXECUTION.
3662  016534  007710                          7710                    ;FPS AFTER EXECUTION.
3663  016536  177777  177777  177777  5$:     .WORD   -1,-1,-1,-1     ;ERROR RES.
3664  016544  177777
3665  016546  104033                  6$:     ERROR   33
3666
3667                                  ;DIVD TEST.
3668  016550                          DDD7:
3669  016550  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3670  016552  004737  016630                  JSR     PC,@#DIVDSUB
3671  016556  170360  170360          1$.     .WORD   170360,170360   ;AC
3672  016562  170360  170360                  .WORD   170360,170360
3673  016566  170360  170360          2$:     .WORD   170360,170360   ;FSRC
3674  016572  170360  170360                  .WORD   170360,170360
3675  016576  040200  000000  000000  3$:     .WORD   40200,0,0,0     ;RES
3676  016604  000000
3677  016606  007717                  4$:     7717                    ;FPS BEFORE EXECUTION.
3678  016610  007700                          7700                    ;FPS AFTER EXECUTION.
3679  016612  177777  177777  177777  5$:     .WORD   -1,-1,-1,-1     ;ERROR RES.
3680  016620  177777
3681  016622  104033                  6$:     ERROR   33
3682
3683  016624  000137  017070                  JMP     @#DDDDONE       ;GO TO NEXT TEST.
3684
3685
3686                                  ;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.

016630  012601                  DIVDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
016632  012700  000200                  MOV     #200,R0         ;SET FD MODE.
016636  170100                  LDFPS   R0

016640  010100                  MOV     R1,R0           ;SET UP THE AC0 OPERAND.
016642  172410                  LDD     (R0),AC0
016644  016100  000030          MOV     30(R1),R0       ;LOAD THE FPS.
016650  170100                  LDFPS   R0

016652  012737  016666  001236          MOV     #1$,@#$TMP2
016660  010100                  MOV     R1,R0           ;ESTABLISH A POINTER TO FSRC.
016662  062700  000010          ADD     #10,R0

016666  174410                  1$:     DIVD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

016670  170204                  STFPS   R4              ;GET THE FPS.
016672  012700  000200          MOV     #200,R0         ;SET FD MODE
016676  170100                  LDFPS   R0

016700  012700  017060          MOV     #DIVDT,R0       ;GET THE RESULT.
016704  174010                  STD     AC0,(R0)

016706  010102                  MOV     R1,R2           ;SAVE DATA IN CASE OF ERROR.
016710  010237  001240          MOV     R2,@#$TMP3
016714  062702  000010          ADD     #10,R2
016720  010237  001242          MOV     R2,@#$TMP4
016724  062702  000010          ADD     #10,R2
016730  010237  001244          MOV     R2,@#$TMP5
016734  012737  017060  001246          MOV     #DIVDT,@#$TMP6
016742  010437  001250          MOV     R4,@#$TMP7
016746  016137  000032  001252          MOV     32(R1),@#$TMP10
```

```
3743
3744  016754  010102                        MOV     R1,R2           ;CHECK THE RESULT.
3745  016756  062702  000020                ADD     #20,R2
3746  016762  012703  017060                MOV     #DIVDT,R3
3747  016766  012705  000004                MOV     #4,R5
3748  016772  022223              2$:       CMP     (R2)+,(R3)+
3749  016774  001006                        BNE     10$             ;BRANCH IF RESULT INCORRECT.
3750  016776  077503                        SOB     R5,2$
3751
3752  017000  026104  000032                CMP     32(R1),R4       ;IS FPS CORRECT?
3753  017004  001023                        BNE     15$             ;BRANCH IF INCORRECT.
3754  017006  000161  000046                JMP     46(R1)          ;RETURN.
3755
3756  017012  010102              10$:      MOV     R1,R2           ;WAS INCORRECT RESULT ANTICIPATED
3757  017014  062702  000034                ADD     #34,R2
3758  017020  012703  017060                MOV     #DIVDT,R3
3759  017024  012705  000004                MOV     #4,R5
3760  017030  022223              11$:      CMP     (R2)+,(R3)+
3761  017032  001005                        BNE     12$             ;BRANCH IF NO.
3762  017034  077503                        SOB     R5,11$
3763  017036  010102                        MOV     R1,R2           ;IF THE INCORRECT RESULT WAS
3764  017040  062702  000044                ADD     #44,R2          ;ANTICIPATED RETURN TO THE
3765                                                                ;ERROR REPORT IN THE CALLING
3766  017044  000112                        JMP     (R2)            ;ROUTINE.
3767
3768  017046              12$:                                      ;REPORT RESULT INCORRECT.
3769  017046  104033              13$:      ERROR   33
3770  017050  000161  000046      14$:      JMP     46(R1)
3771
3772  017054              15$:                                      ;REPORT FPS INCORRECT.
3773  017054  104034              16$:      ERROR   34
3774  017056  000774                        BR      14$
3775
3776  017060  000000  000000  000000  DIVDT:  .WORD   0,0,0,0
3777  017066  000000
3778
3779  017070              DDDDONE:
3780  017070  104412                        RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
3781                                                                ;SEE IF THE USER HAS EXPRESSED
3782                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
3783                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3784                                                                ;THE USER TYPED CONTROL G?).
3785
3786
3787
3788                                        ;;*****************************************************************
3789                                        ;*TEST 10       MULF TEST
3790                                        ;*
3791                                        ;*THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE
3792                                        ;*TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE
3793                                        ;*RESULTS.
3794                                        ;*
3795                                        ;*****************************************************************
3796  017072  000004              TST10:    SCOPE
3797
3798                                        ;MULF WITH (FSRC=AC=0)
```

```
017074                          EEE1:
017074  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
017076  004737  017654                   JSR     PC,@#MULFSUB
017102  000000  000000          1$:     .WORD   0,0             ;AC
017106  000000  000000          2$:     .WORD   0,0             ;FSRC
017112  000000  000000          3$:     .WORD   0,0             ;RES
017116  007517                  4$:     7517                    ;FPS BEFORE EXECUTION.
017120  007504                          7504                    ;FPS AFTER EXECUTION.
017122  177777  177777          5$:     .WORD   -1,-1
017126  104037                  6$:     ERROR   37

                                ;MULF WITH (FSRC=0).
017130                          EEE2:
017130  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
017132  004737  017654                   JSR     PC,@#MULFSUB
017136  071625  034435          1$:     .WORD   71625,34435     ;AC
017142  000000  000000          2$:     .WORD   0,0             ;FSRC
017146  000000  000000          3$:     .WORD   0,0             ;RES
017152  000013                  4$:     13                      ;FPS BEFORE EXECUTION.
017154  000004                          4                       ;FPS AFTER EXECUTION.
017156  177777  177777          5$:     .WORD   -1,-1           ;ERROR RES.
017162  104037                  6$:     ERROR   37

                                ;MULF WITH (AC=0)
017164                          EEE3:
017164  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
017166  004737  017654                   JSR     PC,@#MULFSUB
017172  000000  000000          1$:     .WORD   0,0             ;AC
017176  071625  153443          2$:     .WORD   071625,153443   ;FSRC
017202  000000  000000          3$:     .WORD   0,0             ;RES
017206  007500                  4$:     7500                    ;FPS BEFORE EXECUTION.
017210  007504                          7504                    ;FPS AFTER EXECUTION.
017212  177777  177777          5$:     .WORD   -1,-1           ;ERROR RES.
017216  104037                  6$:     ERROR   37

                                ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
017220                          EEE4:
017220  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
017222  004737  017654                   JSR     PC,@#MULFSUB
017226  040200  000000          1$:     .WORD   40200,0         ;AC
017232  040177  177777          2$:     .WORD   40177,-1        ;FSRC
017236  040177  177777          3$:     .WORD   40177,-1        ;RES
017242  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
017244  000000                          0                       ;FPS AFTER EXECUTION.
017246  140177  177777          5$:     .WORD   140177,-1       ;ERROR RES.
017252  104041                  6$:     ERROR   41              ;BAD SIGN.

                                ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.
017254                          EEE5:
017254  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
017256  004767  000372                   JSR     PC,MULFSUB
017262  040177  177777          1$:     .WORD   40177,-1        ;AC
017266  040200  000000          2$:     .WORD   40200,0         ;FSRC
017272  040177  177777          3$:     .WORD   40177,-1        ;RES
017276  000040                  4$:     40                      ;FPS BEFORE EXECUTION.
017300  000040                          40                      ;FPS AFTER EXECUTION.
```

```
017302  037777  177777          5$:     .WORD   37777,-1            ;ERROR RES.
017306  104042                  6$:     ERROR   42                  ;ST 252 TO 044 INTO 444 (BUT Y62)
                                                                    ;MUL. NORMALIZATION FAILURE.


                                ;MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
017310                          EEE6:
017310  104413                          LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
017312  004737  017654                  JSR     PC,@#MULFSUB
017316  040100  000000          1$:     .WORD   40100,0             ;AC
017322  040100  000000          2$:     .WORD   40100,0             ;FSRC
017326  040020  000000          3$:     .WORD   40020,0             ;RES
017332  000012                  4$:     12                          ;FPS BEFORE EXECUTION.
017334  000000                          0                           ;FPS AFTER EXECUTION.
017336  042040  000000          5$:     .WORD   42040,0             ;ERROR RES.
017342  104043                  6$:     ERROR   43                  ;ST 252 TO 444 INTO 042 (BUT Y62)
                                                                    ;MUL. NORMALIZATION FAILURE.


                                ;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
017344                          EEE7:
017344  104413                          LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
017346  004737  017654                  JSR     PC,@#MULFSUB
017352  017500  000000          1$:     .WORD   17500,0             ;AC
017356  023652  125252          2$:     .WORD   23652,125252        ;FSRC
017362  003177  177777          3$:     .WORD   3177,-1             ;RES
017366  007417                  4$:     7417                        ;FPS BEFORE EXECUTION.
017370  007400                          7400                        ;FPS AFTER EXECUTION.
017372  177777  177777          5$:     .WORD   -1,-1
017376  104037                  6$:     ERROR   37


                                ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
017400                          EEE8:
017400  104413                          LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
017402  004737  017654                  JSR     PC,@#MULFSUB
017406  040342  000000          1$:     .WORD   40342,0             ;AC
017412  176542  000000          2$:     .WORD   176542,0            ;FSRC
017416  176707  102000          3$:     .WORD   176707,102000       ;RES
017422  000007                  4$:     7                           ;FPS BEFORE EXECUTION.
017424  000010                          10                          ;FPS AFTER EXECUTION.
017426  076507  102000          7$:     .WORD   76507,102000        ;ERROR RES.
017432  104041                  6$:     ERROR   41                  ;BAD SIGN.


                                ;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
017434                          EEE9:
017434  104413                          LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
017436  004737  017654                  JSR     PC,@#MULFSUB
017442  140200  000000          1$:     .WORD   140200,0            ;AC
017446  007417  007417          2$:     .WORD   7417,7417           ;FSRC
017452  107417  007417          3$:     .WORD   107417,7417         ;RES
017456  000000                  4$:     0                           ;FPS BEFORE EXECUTION.
017460  000010                          10                          ;FPS AFTER EXECUTION.
017462  007417  007417          5$:     .WORD   7417,7417           ;ERROR RES.
017466  104041                  6$:     ERROR   41                  ;BAD SIGN.


                                ;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
017470                          EEE10:
017470  104413                          LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
```

```
3911  017472  004737  017654                  JSR     PC,@#MULFSUB
3912  017476  144600  000000          1$:     .WORD   144600,0        ;AC
3913  017502  154000  000000          2$:     .WORD   154000,0        ;FSRC
3914  017506  060400  000000          3$:     .WORD   60400,0         ;RES
3915  017512  000017                  4$:     17                      ;FPS BEFORE EXECUTION.
3916  017514  000000                          0                       ;FPS AFTER EXECUTION.
3917  017516  160400  000000          5$:     .WORD   160400,0        ;ERROR RES.
3918  017522  104041                  6$:     ERROR   41              ;BAD SIGN.
3919
3920                                  ;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
3921  017524                          EEE11:
3922  017524  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3923  017526  004737  017654                  JSR     PC,@#MULFSUB
3924  017532  140300  000000          1$:     .WORD   140300,0        ;AC
3925  017536  160000  000001          2$:     .WORD   160000,1        ;FSRC
3926  017542  060100  000002          3$:     .WORD   60100,2         ;RES
3927  017546  000010                  4$:     10                      ;FPS BEFORE EXECUTION.
3928  017550  000000                          0                       ;FPS AFTER EXECUTION.
3929  017552  060100  000001          5$:     .WORD   60100,1         ;ERROR RES.
3930  017556  104044                  6$:     ERROR   44              ;ROUND FAILURE.
3931
3932                                  ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
3933  017560                          EEE12:
3934  017560  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3935  017562  004737  017654                  JSR     PC,@#MULFSUB
3936  017566  060000  000001          1$:     .WORD   60000,1         ;AC
3937  017572  140300  000000          2$:     .WORD   140300,0        ;FSRC
3938  017576  160100  000001          3$:     .WORD   160100,1        ;RES
3939  017602  007547                  4$:     7547                    ;FPS BEFORE EXECUTION.
3940  017604  007550                          7550                    ;FPS AFTER EXECUTION.
3941  017606  160100  000001          5$:     .WORD   160100,1        ;ERROR RES.
3942  017612  104045                  6$:     ERROR   45              ;TRUNCATION ERROR.
3943
3944                                  ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
3945  017614                          EEE13:
3946  017614  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
3947  017616  004737  017654                  JSR     PC,@#MULFSUB
3948  017622  040277  000000          1$:     .WORD   40277,0         ;AC
3949  017626  060000  000001          2$:     .WORD   60000,1         ;FSRC
3950  017632  060077  000001          3$:     .WORD   60077,1         ;RES
3951  017636  000014                  4$:     14                      ;FPS BEFORE EXECUTION.
3952  017640  000000                          0                       ;FPS AFTER EXECUTION.
3953  017642  060077  000002          5$:     .WORD   60077,2         ;ERROR RES.
3954  017646  104044                  6$:     ERROR   44              ;ROUND FAILURE. CONSTANT BAD.
3955
3956  017650  000167  000224                  JMP     EEEDONE         ;GO TO THE NEXT TEST.
3957
3958                                  ;THIS SUBROUTINE, MULFSUB, IS CALLED TO SET UP, EXECUTE
3959                                  ;AND CHECK THE RESULT OF A MULF INSTRUCTION. IT IS CALLED THUS:
3960                                  ;
3961                                  ;               JSR     PC,@#MULFSUB
3962                                  ;       ACARG:  .WORD   X,X             ;AC OPERAND
3963                                  ;       FSRCARG:.WORD   X,X             ;FSRC OPERAND
3964                                  ;       RES:    .WORD   X,X             ;EXPECTED RESULT
3965                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
3966                                  ;       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.

017654  012601                  MULFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
017656  012700  000200                  MOV     #200,R0         ;SET FD MODE.
017662  170100                  LDFPS   R0
017664  010100                  MOV     R1,R0           ;LOAD THE AC OPERAND.
017666  172410                  LDD     (R0),AC0
017670  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
017674  170100                  LDFPS   R0
017676  012737  017712  001236          MOV     #1$,@#$TMP2
017704  010100                  MOV     R1,R0
017706  062700  000004                  ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.

017712  171010          1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.

017714  170204                  STFPS   R4              ;GET THE FPS.
017716  012700  000200                  MOV     #200,R0         ;SET FD MODE
017722  170100                  LDFPS   R0

017724  012700  020070                  MOV     #MULFT,R0       ;GET THE RESULT OF THE MULF.
017730  174010                  STD     AC0,(R0)

017732  010102                  MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
017734  010237  001240                  MOV     R2,@#$TMP3
017740  062702  000004                  ADD     #4,R2
017744  010237  001242                  MOV     R2,@#$TMP4
017750  062702  000004                  ADD     #4,R2
017754  010237  001244                  MOV     R2,@#$TMP5
017760  012737  020070  001246          MOV     #MULFT,@#$TMP6
017766  010437  001250                  MOV     R4,@#$TMP7
017772  016137  000016  001252          MOV     16(R1),@#$TMP10

020000  021061  000010                  CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
020004  001011                  BNE     10$             ;IF INCORRECT BRANCH.
020006  026061  000002  000012          CMP     2(R0),12(R1)
020014  001005                  BNE     10$

020016  026104  000016                  CMP     16(R1),R4       ;IS FPS CORRECT?
020022  001020                  BNE     15$             ;IF INCORRECT BRANCH.
020024  000161  000026                  JMP     26(R1)          ;IF NO ERRORS OCCURRED RETURN.
```

```
4023  020030  021061  000020          10$:    CMP     (R0),20(R1)     ;DOES THE INCORRECT RESULT
4024  020034  001010                          BNE     11$             ;MATCH THE ANTICIPATED INCORRECT RESULT.
4025  020036  026061  000002  000022          CMP     2(R0),22(R1)
4026  020044  001004                          BNE     11$             ;BRANCH IF NO.
4027
4028  020046  010102                          MOV     R1,R2           ;IT MATCHED SO RETURN TO THE ERROR
4029                                                                  ;REPORT AT THE CALLING ROUTINE.
4030  020050  062702  000024                  ADD     #24,R2
4031  020054  000112                          JMP     (R2)
4032
4033  020056                          11$:                            ;REPORT RESULT INCORRECT.
4034  020056  104037                  12$:    ERROR   37
4035  020060  000161  000026          13$:    JMP     26(R1)
4036
4037  020064                          15$:                            ;REPORT FPS INCORRECT.
4038  020064  104040                  16$:    ERROR   40
4039  020066  000774                          BR      13$
4040
4041  020070  000000  000000  000000  MULFT:  .WORD   0,0,0,0
4042  020076  000000
4043
4044  020100                          EEEDONE:
4045  020100  104413                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
4046                                                                  ;SEE IF THE USER HAS EXPRESSED
4047                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
4048                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4049                                                                  ;THE USER TYPED CONTROL G?).
4050
4051
4052
4053                                  ;;**********************************************************************
4054                                  ;*TEST 11       MULD TEST
4055                                  ;*
4056                                  ;*THIS IS A TEST OF THE MULD INSTRUCTION. NOTE THAT A SUBROUTINE IS
4057                                  ;*USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND
4058                                  ;*CHECK THE RESULTS.
4059                                  ;*
4060                                  ;;**********************************************************************
4061  020102  000004                  TST11:  SCOPE
4062
4063                                  ;MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.
4064  020104                          FFF1:
4065  020104  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4066  020106  004737  020370                  JSR     PC,@#MULDSUB
4067  020112  040200  000000  000000  1$:     .WORD   40200,0,0,0     ;AC
4068  020120  000000
4069  020122  023777  177777  177777  2$:     .WORD   23777,-1,-1,-1  ;FSRC
4070  020130  177777
4071  020132  023777  177777  177777  3$:     .WORD   23777,-1,-1,-1  ;RES
4072  020140  177777
4073  020142  000217                  4$:     217                     ;FPS BEFORE EXECUTION.
4074  020144  000200                          200                     ;FPS AFTER EXECUTION.
4075  020146  023777  177777  000000  5$:     .WORD   23777,-1,0,0    ;ERROR RES.
4076  020154  000000
4077  020156  104047                  6$:     ERROR   47              ;BAD CONSTANT USED IN ALGORITHM
4078                                                                  ;USED 24 INSTEAD OF 56.
```

```
4079
4080                                        ;MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
4081  020160                                FFF2:
4082  020160  104413                               LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
4083  020162  004767  000202                      JSR     PC,MULDSUB
4084  020166  065400  000000  000000  1$:         .WORD   65400,0,0,1      ;AC
4085  020174  000001
4086  020176  037577  177777  177777  2$:         .WORD   37577,-1,-1,-2   ;FSRC
4087  020204  177776
4088  020206  064777  177777  177777  3$:         .WORD   64777,-1,-1,-1   ;RES
4089  020214  177777
4090  020216  000247                    4$:       247                      ;FPS BEFORE EXECUTION.
4091  020220  000240                              240                      ;FPS AFTER EXECUTION.
4092  020222  065000  000000  000000  5$:         .WORD   65000,0,0,0      ;ERROR RES.
4093  020230  000000
4094  020232  104050                    6$:       ERROR   50               ;TRUNCATION ERROR.
4095
4096                                        ;MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
4097  020234                                FFF3:
4098  020234  104413                               LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
4099  020236  004737  020370                      JSR     PC,@#MULDSUB
4100  020242  137577  177777  177777  1$:         .WORD   137577,-1,-1,-2  ;AC
4101  020250  177776
4102  020252  165400  000000  000000  2$:         .WORD   165400,0,0,1     ;FSRC
4103  020260  000001
4104  020262  065000  000000  000000  3$:         .WORD   65000,0,0,0      ;RES
4105  020270  000000
4106  020272  007717                    4$:       7717                     ;FPS BEFORE EXECUTION.
4107  020274  007700                              7700                     ;FPS AFTER EXECUTION.
4108  020276  064777  177777  177777  5$:         .WORD   64777,-1,-1,-1   ;ERROR RES.
4109  020304  177777
4110  020306  104051                    6$:       ERROR   51               ;ROUND ERROR.
4111
4112                                        ;MULD   TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
4113  020310                                FFF4:
4114  020310  104413                               LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
4115  020312  004737  020370                      JSR     PC,@#MULDSUB
4116  020316  017500  000000  000000  1$:         .WORD   17500,0,0,0      ;AC
4117  020324  000000
4118  020326  123652  125252            2$:       .WORD   123652,125252    ;FSRC
4119  020332  125252  125252                      .WORD   125252,125252
4120  020336  103177  177777  177777  3$:         .WORD   103177,-1,-1,-1  ;RES
4121  020344  177777
4122  020346  000200                    4$:       200                      ;FPS BEFOR_ EXECUTION.
4123  020350  000210                              210                      ;FPS AFTER EXECUTION.
4124  020352  103200  000000  000000  5$:         .WORD   103200,0,0,0     ;ERROR RES.
4125  020360  000000
4126  020362  104052                    6$:       ERROR   52               ;ROUND ERROR (BAD CONSTANT).
4127
4128  020364  000167  000240                      JMP     FFFDONE
4129
4130                                        ;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
4131                                        ;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:
4132                                        ;
4133                                        ;               JSR     PC,@#MULDSUB
4134                                        ;       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.

020370  012601                  MULDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
020372  012700  000200                  MOV     #200,R0         ;SET FD MODE.
020376  170100                  LDFPS   R0

020400  010100                  MOV     R1,R0           ;SET UP THE AC0 OPERAND.
020402  172410                  LDD     (R0),AC0
020404  016100  000030          MOV     30(R1),R0       ;LOAD THE FPS.
020410  170100                  LDFPS   R0

020412  012737  020426  001236          MOV     #1$,@#$TMP2
020420  010100                  MOV     R1,R0           ;ESTABLISH A POINTER TO FSRC.
020422  062700  000010          ADD     #10,R0

020426  171010          1$:     MULD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

020430  170204                  STFPS   R4              ;GET THE FPS.
020432  012700  000200          MOV     #200,R0         ;SET FD MODE.
020436  170100                  LDFPS   R0

020440  012700  020620          MOV     #MULDT,R0       ;GET THE RESULT.
020444  174010                  STD     AC0,(R0)

020446  010102                  MOV     R1,R2           ;SAVE DATA IN CASE OF ERROR.
020450  010237  001240          MOV     R2,@#$TMP3
020454  062702  000010          ADD     #10,R2
020460  010237  001242          MOV     R2,@#$TMP4
020464  062702  000010          ADD     #10,R2
020470  010237  001244          MOV     R2,@#$TMP5
020474  012737  020620  001246  MOV     #MULDT,@#$TMP6
020502  010437  001250          MOV     R4,@#$TMP7
020506  016137  000032  001252  MOV     32(R1),@#$TMP10

020514  010102                  MOV     R1,R2           ;CHECK THE RESULT.
020516  062702  000020          ADD     #20,R2
020522  012703  020620          MOV     #MULDT,R3
```

```
4191  020526  012705  000004            MOV     #4,R5
4192  020532  022223            2$:     CMP     (R2)+,(R3)+
4193  020534  001006                    BNE     10$             ;BRANCH IF RESULT INCORRECT.
4194  020536  077503                    SOB     R5,2$
4195
4196  020540  026104  000032            CMP     32(R1),R4       ;IS FPS CORRECT?
4197  020544  001023                    BNE     15$             ;BRANCH IF INCORRECT.
4198  020546  000161  000046            JMP     46(R1)          ;RETURN.
4199
4200  020552  010102            10$:    MOV     R1,R2           ;WAS INCORRECT RESULT ANTICIPATED?
4201  020554  062702  000034            ADD     #34,R2
4202  020560  012703  020620            MOV     #MULDT,R3
4203  020564  012705  000004            MOV     #4,R5
4204  020570  022223            11$:    CMP     (R2)+,(R3)+
4205  020572  001005                    BNE     12$             ;BRANCH IF NO.
4206  020574  077503                    SOB     R5,11$
4207  020576  010102                    MOV     R1,R2           ;IF THE INCORRECT RESULT WAS
4208  020600  062702  000044            ADD     #44,R2          ;ANTICIPATED RETURN TO THE
4209                                                            ;ERROR REPORT IN THE CALLING
4210  020604  000112                    JMP     (R2)            ;ROUTINE.
4211
4212  020606                    12$:                            ;REPORT RESULT INCORRECT.
4213  020606  104246            13$:    ERROR   246
4214  020610  000161  000046    14$:    JMP     46(R1)
4215
4216  020614                    15$:                            ;REPORT FPS INCORRECT.
4217  020614  104046            16$:    ERROR   46
4218  020616  000774                    BR      14$
4219
4220  020620  000000  000000  000000  MULDT:  .WORD   0,0,0,0
4221  020626  000000
4222
4223  020630                    FFFDONE:
4224  020630  104412                    RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
4225                                                            ;SEE IF THE USER HAS EXPRESSED
4226                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
4227                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4228                                                            ;THE USER TYPED CONTROL G?).
4229
4230
4231
4232                                    ;;*****************************************************************
4233                                    ;*TEST 12       UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED. TEST
4234                                    ;*
4235                                    ;*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING
4236                                    ;*THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE
4237                                    ;*IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
4238                                    ;*CHECK THE RESULTS.
4239                                    ;*
4240                                    ;*****************************************************************
4241  020632  000004            TST12:  SCOPE
4242
4243                                    ;UNDERFLOW, WITH EXPONENT OF RESULT = -129
4244  020634                    1111:
4245  020634  104413                    LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4246  020636  004737  021060            JSR     PC,@#OVUNFNT
```

```
 4247  020642  020200  000000          1$:     .WORD   20200,0         ;AC
 4248  020646  020000  000000          2$:     .WORD   20000,0         ;FSRC
 4249  020652  000000  000000          3$:     .WORD   0,0             ;RES
 4250  020656  177777  177777          4$:     .WORD   -1,-1           ;ERROR RES.
 4251  020662  000000                  5$:     0                       ;FPS BEFORE EXECUTION.
 4252  020664  000004                          4                       ;FPS AFTER EXECUTION.
 4253  020666  000012                  6$:     12                      ;FEC
 4254  020670  177777                          -1                      ;FLAG
 4255  020672  104117                  7$:     ERROR   117             ;ST 331 TO 155 INTO 115 (BUT FIU)
 4256  020674  000401                          BR      8$
 4257  020676  104114                          ERROR   114
 4258  020700                          8$:
 4259
 4260                                  ;UNDERFLOW, WITH EXPONENT OF RESULT = -193
 4261  020700                          1112:
 4262  020700  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
 4263  020702  004737  021060                  JSR     PC,@#OVUNFNT
 4264  020706  010200  000000          1$:     .WORD   10200,0         ;AC
 4265  020712  010000  000000          2$:     .WORD   10000,0         ;FSRC
 4266  020716  000000  000000          3$:     .WORD   0,0             ;RES
 4267  020722  010000  000000          4$:     .WORD   10000,0         ;ERROR RES.
 4268  020726  005013                  5$:     5013                    ;FPS BEFORE EXECUTION.
 4269  020730  005004                          5004                    ;FPS AFTER EXECUTION.
 4270  020732  000012                  6$:     12                      ;FEC
 4271  020734  177777                          -1                      ;FLAG
 4272  020736  104120                  7$:     ERROR   120             ;SETTING FIUV OR FIV CAUSES TRAP
 4273                                                                  ;WITH FIU CLEAR.
 4274  020740  000401                          BR      8$
 4275  020742  104114                          ERROR   114
 4276  020744                          8$:
 4277
 4278                                  ;OVERFLOW, EXPONENT OF RESULT = 128
 4279  020744                          1113:
 4280  020744  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
 4281  020746  004737  021060                  JSR     PC,@#OVUNFNT
 4282  020752  060200  000000          1$:     .WORD   60200,0         ;AC
 4283  0207    [illegible]  000000     2$:     .WORD   60000,0         ;FSRC
 4284  0207    [illegible]  000000     3$:     .WORD   0,0             ;RES
 4285  0207E6  06[illegible]  000000   4$:     .WORD   60000,0         ;ERROR RES.
 4286  020772  000000                  5$:     0                       ;FPS BEFORE EXECUTION.
 4287  0207    000006                          6                       ;FPS AFTER EXECUTION.
 4288  020776  000010                  6$:     10                      ;FEC
 4289  021000  000000                          0                       ;FLAG
 4290  021002  104121                  7$:     ERROR   121             ;ST 333 TO 136 INTO 116 (BUT FIV).
 4291  021004  000401                          BR      8$
 4292  021006  104113                          ERROR   113
 4293  021010                          8$:
 4294
 4295                                  ;OVERFLOW, EXPONENT OF RESULT = 130
 4296  021010                          1114:
 4297  021010  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
 4298  021012  004737  021060                  JSR     PC,@#OVUNFNT
 4299  021016  060200  000000          1$:     .WORD   60200,0         ;AC
 4300  021022  060200  000000          2$:     .WORD   60200,0         ;FSRC
 4301  021026  000000  000000          3$:     .WORD   0,0             ;RES
 4302  021032  177777  177777          4$:     .WORD   -1,-1           ;ERROR RES.
```

```
                                        5$:     6011                    ;FPS BEFORE EXECUTION.
                                                6006                    ;FPS AFTER EXECUTION.
                                        6$:     10                      ;FEC
                                                0                       ;FLAG
                                        7$:     ERROR   122             ;SETTING FIUV OR FIU WITH
                                                                        ;FIV CLEAR CAUSES TRAP.
                                                BR      8$
                                                ERROR   113
                                        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).

021060  012601                  OVUNFNT:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
021062  012700  000200                  MOV     #200,R0         ;SET FD MODE.
021066  170100                  LDFPS   R0

021070  010100                  MOV     R1,R0           ;LOAD AC0, OPERAND.
021072  172410                  LDD     (R0),AC0
```

```
4359
4360  021074  010102                          MOV     R1,R2           ;SAVE THE DATA PATTERNS IN CASE OF
4361  021076  010237  001240                  MOV     R2,@#$TMP3      ;ERROR.
4362  021102  062702  000004                  ADD     #4,R2
4363  021106  010237  001242                  MOV     R2,@#$TMP4
4364  021112  062702  000004                  ADD     #4,R2
4365  021116  010237  001244                  MOV     R2,@#$TMP5
4366  021122  016137  000022  001252          MOV     22(R1),@#$TMP10
4367  021130  012737  021460  001246          MOV     #OVFNTT,@#$TMP6
4368
4369  021136  016100  000020                  MOV     20(R1),R0       ;LOAD THE FPS.
4370  021142  170100                          LDFPS   R0
4371  021144  012737  021166  001236          MOV     #1$,@#$TMP2
4372  021152  012737  021352  000244          MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
4373                                                                  ;OF ERROR.
4374  021160  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
4375  021162  062700  000004                  ADD     #4,R0
4376
4377  021166  171010                  1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.
4378
4379  021170  170204                  2$:     STFPS   R4              ;GET FPS.
4380  021172  170305                          STST    R5              ;GET FEC.
4381  021174  012700  000200                  MOV     #200,R0         ;SET FD MODE.
4382  021200  170100                          LDFPS   R0
4383  021202  012700  021460                  MOV     #OVFNTT,R0      ;GET THE RESULT.
4384  021206  174010                          STD     AC0,(R0)
4385  021210  010437  001250                  MOV     R4,@#$TMP7
4386  021214  010537  001254                  MOV     R5,@#$TMP11
4387
4388  021220  012700  021460                  MOV     #OVFNTT,R0      ;CHECK THE RESULT.
4389  021224  010102                          MOV     R1,R2
4390  021226  062702  000010                  ADD     #10,R2
4391  021232  012703  000002                  MOV     #2,R3
4392  021236  022022                  3$:     CMP     (R0)+,(R2)+
4393  021240  001015                          BNE     15$             ;BRANCH IF INCORRECT.
4394  021242  077303                          SOB     R3,3$
4395
4396  021244  026104  000022                  CMP     22(R1),R4       ;WAS FPS CORRECT?
4397  021250  001002                          BNE     10$             ;BRANCH IF FPS IS INCORRECT.
4398
4399  021252  000161  000036          4$:     JMP     36(R1)          ;RETURN, TEST COMPLETED.
4400
4401                                  ;REPORT INCORRECT FPS.
4402  021256  005761  000026          10$:    TST     26(R1)          ;WAS THE RESULT OVER OR UNDER FLOW?
4403  021262  001002                          BNE     12$             ;BRANCH IF UNDERFLOW.
4404
4405                                                                  ;REPORT FPS BAD AFTER OVERFLOW.
4406  021264  104111                  11$:    ERROR   111
4407  021266  000771                          BR      4$
4408
4409  021270                          12$:                            ;REPORT FPS BAD AFTER UNDERFLOW.
4410  021270  104112                  13$:    ERROR   112
4411  021272  000767                          BR      4$
4412
4413                                  ;RESULT INCORRECT.
4414  021274  012700  021460          15$:    MOV     #OVFNTT,R0      ;SEE IF FAILURE IS ANTICIPATED
```

```
4415  021300  010102                      MOV     R1,R2           ;FAILURE.
4416  021302  062702  000014              ADD     #14,R2
4417  021306  012703  000002              MOV     #2,R3
4418  021312  022022              16$:    CMP     (R0)+,(R2)+
4419  021314  001007                      BNE     17$             ;BRANCH IF NOT ANTICIPATED.
4420  021316  077303                      SOB     R3,16$
4421
4422  021320  010102                      MOV     R1,R2           ;ERROR WAS ANTICIPATED SO RETURN
4423  021322  062702  000034              ADD     #34,R2          ;TO THE ERROR REPORT IN THE CALLING
4424  021326  010237  001236              MOV     R2,@#$TMP2      ;ROUTINE.
4425  021332  000112                      JMP     (R2)
4426
4427  021334  005761  000026      17$:    TST     26(R1)          ;RESULT WAS NOT ANTICIPATED
4428                                                              ;SO ERROR MUST BE REPORTED HERE.
4429                                                              ;FIRST SEE IF ARGUMENTS SHOULD
4430                                                              ;HAVE RESULTED IN OVERFLOW OR UNDER
4431                                                              ;FLOW BY LOOKING AT THE FLAG.
4432  021340  001002                      BNE     19$             ;BRANCH IF UNDERFLOW EXPECTED.
4433
4434                                                              ;REPORT RESULT INCORRECT, EXPECTING
4435  021342  104113              18$:    ERROR   113             ;OVERFLOW.
4436  021344  000742                      BR      4$
4437
4438  021346                      19$:                            ;REPORT RESULT INCORRECT, EXPECTING
4439  021346  104114              20$:    ERROR   114             ;UNDERFLOW.
4440  021350  000740                      BR      4$
4441
4442                              ;IF AN FP TRAP OCCURS COME HERE.
4443  021352  011602              25$.    MOV     (SP),R2         ;GET ADDRESS OF TRAP.
4444  021354  022702  021170              CMP     #2$,R2          ;WAS THE TRAP DURING THE MULF INSTRUCTION?
4445  021360  001402                      BEQ     26$             ;BRANCH IF YES.
4446  021362  000137  036614              JMP     @#FPSPUR        ;OTHERWISE GO REPORT A SPURIOUS
4447                                                              ;FP TRAP.
4448  021366  022626              26$:    CMP     (SP)+,(SP)+     ;RESET THE STACK.
4449  021370  010237  001236              MOV     R2,@#$TMP2      ;SAVE DATA FOR ERROR REPORT.
4450  021374  170204                      STFPS   R4              ;GET FPS.
4451  021376  170305                      STST    R5              ;GET FEC.
4452  021400  012700  000200              MOV     #200,R0         ;SET FD MODE.
4453  021404  170100                      LDFPS   R0
4454  021406  012700  021460              MOV     #OVFNTT,R0      ;GET THE RESULT.
4455  021412  174010                      STD     AC0,(R0)
4456  021414  010537  001254              MOV     R5,@#$TMP11
4457  021420  020561  000024              CMP     R5,24(R1)       ;WAS THE FEC ANTICIPATED?
4458  021424  001004                      BNE     27$             ;BRANCH IF NOT ANTICIPATED.
4459
4460  021426  010102                      MOV     R1,R2           ;ERROR WAS ANTICIPATED SO
4461  021430  062702  000030              ADD     #30,R2          ;RETURN TO THE ERROR REPORT OF THE
4462                                                              ;CALLING ROUTINE.
4463  021434  000112                      JMP     (R2)
4464
4465  021436  005761  000026      27$:    TST     26(R1)          ;THE ERROR WAS NOT ANTICIPATED SO
4466                                                              ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
4467                                                              ;OVERFLOW OR UNDER FLOW.
4468  021442  001003                      BNE     29$             ;BRANCH IF EXPECTING UNDERFLOW
4469
4470                                                              ;REPORT TRAPPED ON OVERFLOW WITH FIV=0
```

```
4471  021444  104115                  28$:    ERROR   115
4472  021446  000161  000036                  JMP     36(R1)
4473
4474  021452                          29$:                            ;REPORT TRAPPED ON UNDER FLOW WITH FIXED
4475  021452  104116                  30$:    ERROR   116
4476  021454  000161  000036                  JMP     36(R1)
4477
4478  021460  000000  000000  000000  OVFNTT: .WORD   0,0,0,0
4479  021466  000000
4480
4481  021470                          IIIDONE:
4482  021470  104412                          RSETJP                  ;GO INITIALIZE THE FPS AND STACK; AND
4483                                                                  ;SEE IF THE USER HAS EXPRESSED
4484                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
4485                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER .HAS
4486                                                                  ;THE USER TYPED CONTROL G?..
4487
4488
4489
4490
4491                                    ;;*****************************************************************
4492                                    ;*TEST 13       UNDER\OVER FLOW, USING MULD WITH TRAP DISABLED, TEST
4493                                    ;*
4494                                    ;*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN
4495                                    ;*ARRISE USING THE MULD INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS
4496                                    ;*USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND
4497                                    ;*CHECK THE RESULTS.
4498                                    ;*
4499                                    ;*****************************************************************
4500  021472  000004                  TST13:  SCOPE
4501
4502                                    ;UNDERFLOW, EXPONENT OF RESULT=-129
4503  021474                          JJJ1:
4504  021474  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4505  021476  004737  022020                  JSR     PC,@#OVUNDNT
4506  021502  020200  000000          1$:     .WORD   20200,0         ;AC
4507  021506  127272  000000                  .WORD   127272,0
4508  021512  020000  000000  000000  2$:     .WORD   20000,0,0,0     ;FSRC
4509  021520  000000
4510  021522  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
4511  021530  000000
4512  021532  000000  000000          4$:     .WORD   0,0             ;ERROR RES.
4513  021536  127272  000000                  .WORD   127272,0
4514  021542  000200                  5$:     200                     ;FPS BEFORE EXECUTION.
4515  021544  000204                          204                     ;FPS AFTER EXECUTION.
4516  021546  000012                  6$:     12                      ;FEC
4517  021550  177777                          -1                      ;FLAG
4518  021552  104131                  7$:     ERROR   131             ;ST 331 TO 155 INTO 115 (BUT FIU)
4519  021554  000401                          BR      8$
4520  021556  104132                          ERROR   132             ;ST 115 (BUT FD)
4521  021560                          8$:
4522
4523                                    ;UNDERFLOW, EXPONENT OF RESULT = -193
4524  021560                          JJJ2:
4525  021560  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4526  021562  004737  022020                  JSR     PC,@#OVUNDNT
```

```
                                        T13     UNDER\OVER FLOW, USING MULD WITH TRAP DISABLED. TEST

4527  021566  010200  000000          1$:     .WORD   10200,0         ;AC
4528  021572  123456  000000                  .WORD   123456,0
4529  021576  010000  000000  000000  2$:     .WORD   10000,0,0,0     ;FSRC
4530  021604  000000
4531  021606  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
4532  021614  000000
4533  021616  000000  000000  123456  4$:     .WORD   0,0,123456,0    ;ERROR RES
4534  021624  000000
4535  021626  005213                  5$:     5213                    ;FPS BEFORE EXECUTION.
4536  021630  005204                          5204                    ;FPS AFTER EXECUTION.
4537  021632  000012                  6$:     12                      ;FEC
4538  021634  177777                          -1                      ;FLAG
4539  021636  104133                  7$:     ERROR   133             ;SETTING FIUV OR FIV BAD.
4540  021640  000401                          BR      8$
4541  021642  104132                          ERROR   132             ;ST 115 (BUT FD)
4542  021644                          8$:
4543
4544                                  ;OVERFLOW, EXPONENT OF RESULT = 128
4545  021644                          JJJ3:
4546  021644  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4547  021646  004737  022020                  JSR     PC,@#OVUNDNT
4548  021652  060200  000000          1$:     .WORD   60200,0         ;AC
4549  021656  065432  000000                  .WORD   65432,0
4550  021662  060000  000000  000000  2$:     .WORD   60000,0,0,0     ;FSRC
4551  021670  000000
4552  021672  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
4553  021700  000000
4554  021702  000000  000000  065432  4$:     .WORD   0,0,65432,0     ;ERROR RES.
4555  021710  000000
4556  021712  000200                  5$:     200                     ;FPS BEFORE EXECUTION.
4557  021714  000206                          206                     ;FPS AFTER EXECUTION.
4558  021716  000010                  6$:     10                      ;FEC
4559  021720  000000                          0                       ;FLAG
4560  021722  104134                  7$:     ERROR   134             ;ST 333 TO 136 INTO 116 (BUT FIV)
4561  021724  000401                          BR      8$
4562  021726  104135                          ERROR   135             ;ST 116 (BUT FD)
4563  021730                          8$:
4564
4565
4566                                  ;OVERFLOW, EXPONENT OF RESULT = 130
4567  021730                          JJJ4:
4568  021730  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
4569  021732  004737  022020                  JSR     PC,@#OVUNDNT
4570  021736  060200  000000          1$:     .WORD   60200,0         ;AC
4571  021742  125252  000000                  .WORD   125252,0
4572  021746  060200  000000  000000  2$:     .WORD   60200,0,0,0     ;FSRC
4573  021754  000000
4574  021756  000000  000000  000000  3$:     .WORD   0,0,0,0         ;RES
4575  021764  000000
4576  021766  000000  000000  125252  4$:     .WORD   0,0,125252,0    ;ERROR RES.
4577  021774  000000
4578  021776  006211                  5$:     6211                    ;FPS BEFORE EXECUTION.
4579  022000  006206                          6206                    ;FPS AFTER EXECUTION.
4580  022002  000010                  6$:     10                      ;FEC
4581  022004  000000                          0                       ;FLAG
4582  022006  104136                  7$:     ERROR   136             ;SETTING FIUV OR FIV BAD.
```

```
                                        BR      B$
                                        ERROR   135             ;ST 116 (BUT FD)
022014  000137  022430        B$:     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).

022020  012601          OVUNDNT:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
022022  012700  000200                MOV     #200,R0         ;SET FD MODE.
022026  170100                        LDFPS   R0

022030  010100                        MOV     R1,R0           ;LOAD AC0, OPERAND.
022032  172410                        LDD     (R0),AC0

022034  010102                        MOV     R1,R2           ;SAVE THE DATA PATTERNS IN CASE OF
022036  010237  001240                MOV     R2,@#$TMP3      ;ERROR.
022042  062702  000010                ADD     #10,R2
022046  010237  001242                MOV     R2,@#$TMP4
022052  062702  000010                ADD     #10,R2
022056  010237  001244                MOV     R2,@#$TMP5
022062  016137  000042  001252        MOV     42(R1),@#$TMP10
```

```
4639  022070  012737  022420  001246          MOV     #OVDNTT,@#$TMP6
4640
4641  022076  016100  000040                  MOV     40(R1),R0       ;LOAD THE FPS.
4642  022102  170100                          LDFPS   R0
4643  022104  012737  022126  001236          MOV     #1$,@#$TMP2
4644  022112  012737  022312  000244          MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
4645                                                                  ;OF ERROR.
4646  022120  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
4647  022122  062700  000010                  ADD     #10,R0
4648
4649  022126  171010                  1$:     MULD    (R0),AC0        ;TEST INSTRUCTION.
4650
4651  022130  170204                  2$:     STFPS   R4              ;GET FPS.
4652  022132  170305                          STST    R5              ;GET FEC.
4653  022134  012700  000200                  MOV     #200,R0         ;SET FD MODE.
4654  022140  170100                          LDFPS   R0
4655  022142  012700  022420                  MOV     #OVDNTT,R0      ;GET THE RESULT.
4656  022146  174010                          STD     AC0,(R0)
4657  022150  010437  001250                  MOV     R4,@#$TMP7
4658  022154  010537  001254                  MOV     R5,@#$TMP11
4659
4660  022160  012700  022420                  MOV     #OVDNTT,R0      ;CHECK THE RESULT.
4661  022164  010102                          MOV     R1,R2
4662  022166  062702  000020                  ADD     #20,R2
4663  022172  012703  000004                  MOV     #4,R3
4664  022176  022022                  3$:     CMP     (R0)+,(R2)+
4665  022200  001015                          BNE     15$             ;BRANCH IF INCORRECT.
4666  022202  077303                          SOB     R3,3$
4667
4668  022204  026104  000042                  CMP     42(R1),R4       ;WAS FPS CORRECT?
4669  022210  001002                          BNE     10$             ;BRANCH IF FPS IS INCORRECT.
4670
4671  022212  000161  000056          4$:     JMP     56(R1)          ;RETURN, TEST COMPLETED.
4672
4673                                  ;REPORT INCORRECT FPS.
4674  022216  005761  000046          10$:    TST     46(R1)          ;WAS THE RESULT OVER OR UNDER FLOW?
4675  022222  001002                          BNE     12$             ;BRANCH IF UNDERFLOW.
4676
4677                                                                  ;REPORT FPS BAD AFTER OVERFLOW.
4678  022224  104123                  11$:    ERROR   123
4679  022226  000771                          BR      4$
4680
4681  022230                          12$:                            ;REPORT FPS BAD AFTER UNDERFLOW.
4682  022230  104124                  13$:    ERROR   124
4683  022232  000767                          BR      4$
4684
4685                                  ;RESULT INCORRECT.
4686  022234  012700  022420          15$:    MOV     #OVDNTT,R0      ;SEE IF FAILURE IS ANTICIPATED
4687  022240  010102                          MOV     R1,R2           ;FAILURE.
4688  022242  062702  000030                  ADD     #30,R2
4689  022246  012703  000004                  MOV     #4,R3
4690  022252  022022                  16$:    CMP     (R0)+,(R2)+
4691  022254  001007                          BNE     17$             ;BRANCH IF NOT ANTICIPATED.
4692  022256  077303                          SOB     R3,16$
4693
4694  022260  010102                          MOV     R1,R2           ;ERROR WAS ANTICIPATFD SO RETURN
```

```
 4695  022262  062702  000054              ADD     #54,R2          ;TO THE ERROR REPORT IN THE CALLING
 4696  022266  010237  001236              MOV     R2,@#$TMP2      ;ROUTINE.
 4697  022272  000112                      JMP     (R2)
 4698
 4699  022274  005761  000046  17$:        TST     46(R1)          ;RESULT WAS NOT ANTICIPATED
 4700                                                              ;SO ERROR MUST BE REPORTED HERE.
 4701                                                              ;FIRST SEE IF ARGUMENTS SHOULD
 4702                                                              ;HAVE RESULTED IN OVERFLOW OR UNDER
 4703                                                              ;FLOW BY LOOKING AT THE FLAG.
 4704  022300  001002                      BNE     19$             ;BRANCH IF UNDERFLOW EXPECTED.
 4705
 4706                                                              ;REPORT RESULT INCORRECT, EXPECTING
 4707  022302  104125          18$:        ERROR   125             ;OVERFLOW.
 4708  022304  000742                      BR      4$
 4709
 4710  022306                  19$:                                ;REPORT RESULT INCORRECT, EXPECTING
 4711  022306  104126          20$:        ERROR   126             ;UNDERFLOW.
 4712  022310  000740                      BR      4$
 4713
 4714                          ;IF AN FP TRAP OCCURS COME HERE.
 4715  022312  011602          25$:        MOV     (SP),R2         ;GET ADDRESS OF TRAP.
 4716  022314  022702  022130              CMP     #2$,R2          ;WAS THE TRAP DURING THE MULF INSTRUCTION
 4717  022320  001402                      BEQ     26$             ;BRANCH IF YES.
 4718  022322  000137  036614              JMP     @#FPSPUR        ;OTHERWISE GO REPORT A SPURIOUS
 4719                                                              ;FP TRAP.
 4720  022326  022626          26$:        CMP     (SP)+,(SP)+     ;RESET THE STACK.
 4721  022330  010237  001236              MOV     R2,@#$TMP2      ;SAVE DATA FOR ERROR REPORT.
 4722  022334  170204                      STFPS   R4              ;GET FPS.
 4723  022336  170305                      STST    R5              ;GET FEC.
 4724  022340  012700  000200              MOV     #200,R0         ;SET FD MODE.
 4725  022344  170100                      LDFPS   R0
 4726  022346  012700  022420              MOV     #OVDNTT,R0      ;GET THE RESULT.
 4727  022352  174010                      STD     AC0,(R0)
 4728  022354  010537  001254              MOV     R5,@#$TMP11
 4729  022360  020561  000044              CMP     R5,44(R1)       ;WAS THE FEC ANTICIPATED?
 4730  022364  001004                      BNE     27$             ;BRANCH IF NOT ANTICIPATED.
 4731
 4732  022366  010102                      MOV     R1,R2           ;ERROR WAS ANTICIPATED SO
 4733  022370  062702  000050              ADD     #50,R2          ;RETURN TO THE ERROR REPORT OF THE
 4734                                                              ;CALLING ROUTINE.
 4735  022374  000112                      JMP     (R2)
 4736
 4737  022376  005761  000026  27$:        TST     26(R1)          ;THE ERROR WAS NOT ANTICIPATED SO
 4738                                                              ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
 4739                                                              ;OVERFLOW OR UNDER FLOW.
 4740  022402  001003                      BNE     29$             ;BRANCH IF EXPECTING UNDERFLOW
 4741
 4742                                                              ;REPORT TRAPPED ON OVERFLOW WITH FIV=0
 4743  022404  104127          28$:        ERROR   127
 4744  022406  000161  000056              JMP     56(R1)
 4745
 4746  022412                  29$:                                ;REPORT TRAPPED ON UNDER FLOW WITH FIU=0
 4747  022412  104130          30$:        ERROR   130
 4748  022414  000161  000056              JMP     56(R1)
 4749
 4750  022420  000000  000000  000000  OVDNTT: .WORD   0,0,0,0
```

```
 4751  022426  000000
 4752
 4753  022430                        JJJDONE:
 4754  022430  104412                        RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
 4755                                                                ;SEE IF THE USER HAS EXPRESSED
 4756                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
 4757                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (WAS
 4758                                                                ;THE USER TYPED CONTROL G?).
 4759
 4760
 4761
 4762
 4763                                ;;******************************************************************
 4764                                ;*TEST 14       UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED. TEST
 4765                                ;*
 4766                                ;*THIS IS A TEST OF THE UNDERFLOW AND OVERFLOW
 4767                                ;*CONDITIONS THAT CAN OCCUR USING THE MULF INSTRUCTION.
 4768                                ;*A SUBROUTINE IS CALLED TO SET UP THE OPERANDS,
 4769                                ;* EXECUTE THE MULF INSTRUCTION AND CHECK
 4770                                ;*THE RESULTS. HERE THE PARTICULAR INTERRUPT,
 4771                                ;*EITHER OVERFLOW OR UNDERFLOW, IS ENABLED SO A TRAP SHOULD
 4772                                ;*OCCUR.
 4773                                ;*
 4774                                ;;******************************************************************
 4775  022432  000004                TST14:  SCOPE
 4776
 4777                                ;UNDERFLOW, EXPONENT OF RESULT = -129
 4778  022434                        KKK1:
 4779  022434  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
 4780  022436  004737  022660                JSR     PC,@#OVUNFT
 4781  022442  020123  045676        1$:     .WORD   20123,45676     ;AC
 4782  022446  020200  000000        2$:     .WORD   20200,0         ;FSRC
 4783  022452  000123  045676        3$:     .WORD   123,45676       ;RES
 4784  022456  177777  177777        4$:     .WORD   -1,-1           ;ERROR RES.
 4785  022462  002000                5$:     2000                    ;FPS BEFORE EXECUTION.
 4786  022464  102004                        102004                  ;FPS AFTER EXECUTION.
 4787  022466  000012                6$:     12                      ;FEC
 4788  022470  177777                        -1                      ;FLAG
 4789  022472  104145                7$:     ERROR   145             ;ST 331 (BUT FIU) NO TRAP.
 4790  022474  000401                        BR      8$
 4791  022476  104144                        ERROR   144
 4792  022500                        8$:
 4793
 4794                                ;UNDERFLOW, EXPONENT OF THE RESULT = -193
 4795  022500                        KKK3:
 4796  022500  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
 4797  022502  004737  022660                JSR     PC,@#OVUNFT
 4798  022506  010127  127272        1$:     .WORD   10127,127272    ;AC
 4799  022512  010200  000000        2$:     .WORD   10200,0         ;FSRC
 4800  022516  060127  127272        3$:     .WORD   60127,127272    ;RES
 4801  022522  177777  177777        4$:     .WORD   -1,-1           ;ERROR RES.
 4802  022526  007017                5$:     7017                    ;FPS BEFORE EXECUTION.
 4803  022530  107000                        107000                  ;FPS AFTER EXECUTION.
 4804  022532  000012                6$:     12                      ;FEC
 4805  022534  177777                        -1
 4806  022536  104146                7$:     ERROR   146             ;ST 137 (BUT FIJ) NO TRAP.
```

```
 4807  022540  000401                        BR      8$
 4808  022542  104144                        ERROR   144
 4809  022544                        8$:
 4810
 4811                                ;OVERFLOW, EXPONENT OF THE RESULT = 128
 4812  022544                        KKK4:
 4813  022544  104413                        LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
 4814  022546  004737  022660                JSR     PC,@#OVUNFT
 4815  022552  060252  125252        1$:     .WORD   60252,125252        ;AC
 4816  022556  060000  000000        2$:     .WORD   60000,0             ;FSRC
 4817  022562  000052  125252        3$:     .WORD   000052,125252       ;RES
 4818  022566  177777  177777        4$:     .WORD   -1,-1               ;ERROR RES.
 4819  022572  001000                5$:     1000                        ;FPS BEFORE EXECUTION.
 4820  022574  101006                        101006                      ;FPS AFTER EXECUTION.
 4821  022576  000010                6$:     10                          ;FEC
 4822  022600  000000                        0                           ;FLAG
 4823  022602  104147                7$:     ERROR   147                 ;ST 333 (BUT FIV) NO TRAP
 4824  022604  000401                        BR      8$
 4825  022606  104143                        ERROR   143
 4826  022610                        8$:
 4827
 4828                                ;OVERFLOW, EXPONENT OF RESULT = 130
 4829  022610                        KKK5:
 4830  022610  104413                        LPERR                       ;SET UP THE LOOP ON ERROR ADDRESS.
 4831  022612  004737  022660                JSR     PC,@#OVUNFT
 4832  022616  060345  067654        1$:     .WORD   60345,67654         ;AC
 4833  022622  060200  000000        2$:     .WORD   60200,0             ;FSRC
 4834  022626  000345  067654        3$:     .WORD   345,67654           ;RES
 4835  022632  177777  177777        4$:     .WORD   -1,-1               ;ERROR RES.
 4836  022636  007015                5$:     7015                        ;FPS BEFORE EXECUTION.
 4837  022640  107002                        107002                      ;FPS AFTER EXECUTION.
 4838  022642  000010                6$:     10                          ;FEC
 4839  022644  000000                        0                           ;FLAG
 4840  022646  104150                7$:     ERROR   150                 ;ST 133 (BUT FIV) NO TRAP
 4841  022650  000401                        BR      6$
 4842  022652  104143                        ERROR   143
 4843  022654  000167  000412        8$:     JMP     KKKDONE
 4844
 4845                                ;THIS SUBROUTINE, OVUNFT, IS USED TO SET UP THE OPERANDS, EXECUTE
 4846                                ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
 4847                                ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
 4848                                ;TO IT IS MADE THUS:
 4849                                ;
 4850                                ;               ACARG:  .WORD   X,X             ;AC OPERAND
 4851                                ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
 4852                                ;               RES:    .WORD   X,X             ;EXPECTED RESULT
 4853                                ;               ERRES:  .WORD   X,X             ;ERROR RESULT
 4854                                ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
 4855                                ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
 4856                                ;               FEC:    .WORD   X               ;EXPECTED FEC
 4857                                ;               FLAG:   .WORD   X               ;0,-1,OVER/UNDER FLOW FLAG
 4858                                ;               ERR1:   ERROR   X               ;TRAP ERROR.
 4859                                ;               BR      CONT
 4860                                ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
 4861                                ;               CONT:                           ;RETURN ADDRESS
 4862                                ;
```

```
                                ;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 FPSP 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).

022660  012601          OVUNFT: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
022662  012700  000200          MOV     #200,R0         ;SET FD MODE.
022666  170100                  LDFPS   R0

022670  010100                  MOV     R1,R0           ;LOAD AC0, OPERAND.
022672  172410                  LDD     (R0),AC0

022674  010102                  MOV     R1,R2           ;SAVE THE DATA PATTERNS IN CASE OF
022676  010237  001240          MOV     R2,@#$TMP3      ;ERROR.
022702  062702  000004          ADD     #4,R2
022706  010237  001242          MOV     R2,@#$TMP4
022712  062702  000004          ADD     #4,R2
022716  010237  001244          MOV     R2,@#$TMP5
022722  016137  000022  001252  MOV     22(R1),@#$TMP10
022730  012737  023262  001246  MOV     #OVFTT,@#$TMP6

022736  016100  000020          MOV     20(R1),R0       ;LOAD THE FPS.
022742  170100                  LDFPS   R0
022744  012737  022766  001236  MOV     #1$,@#$TMP2
022752  012737  022776  000244  MOV     #50$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
                                                        ;OF ERROR.
022760  010100                  MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
022762  062700  000004          ADD     #4,R0

022766  171010          1$:     MULF    (R0),AC0        ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
022770  170000          2$:     CFCC

022772  000137  023222          JMP     @#25$           ;FAILURE, NO TRAP.

022776  011602          50$:    MOV     (SP),R2         ;TRAP TO HERE AND SEE IF THE PC OF THE
023000  020227  022770          CMP     R2,#2$          ;TRAP WAS THAT OF THE MULF INSTRUCTION.
023004  001402                  BEQ     51$             ;BRANCH IF YES.
023006  000137  036614          JMP     @#FPSPUR        ;OTHERWISE REPORT SPURIOUS FP ERROR.

023012  022626          51$:    CMP     (SP)+,(SP)+     ;RESET THE STACK
023014  170204                  STFPS   R4              ;GET FPS.
023016  170305                  STST    R5              ;GET FEC.
023020  012700  000200          MOV     #200,R0         ;SET FD MODE.
```

```
 4919  023024  170100                        LDFPS   R0
 4920  023026  012700  023262                MOV     #OVFTT,R0       ;GET THE RESULT.
 4921  023032  174010                        STD     AC0,(R0)
 4922  023034  010437  001250                MOV     R4,@#$TMP7
 4923  023040  010537  001254                MOV     R5,@#$TMP11
 4924
 4925  023044  012700  023262                MOV     #OVFTT,R0       ;CHECK THE RESULT.
 4926  023050  010102                        MOV     R1,R2
 4927  023052  062702  000010                ADD     #10,R2
 4928  023056  012703  000002                MOV     #2,R3
 4929  023062  022022              3$:       CMP     (R0)+,(R2)+
 4930  023064  001027                        BNE     15$             ;BRANCH IF INCORRECT.
 4931  023066  077303                        SOB     R3,3$
 4932
 4933  023070  026104  000022                CMP     22(R1),R4       ;WAS FPS CORRECT?
 4934  023074  001014                        BNE     10$             ;BRANCH IF FPS IS INCORRECT.
 4935
 4936  023076  026105  000024                CMP     24(R1),R5       ;IS FEC CORRECT?
 4937  023102  001002                        BNE     5$              ;IF INCORRECT BRANCH.
 4938  023104  000161  000036        4$:     JMP     36(R1)          ;RETURN, TEST COMPLETED.
 4939
 4940                                  ;REPORT INCORRECT FEC.
 4941  023110  005761  000026        5$:     TST     26(R1)          ;WAS THE RESULT OVERFLOW OR UNDERFLOW?
 4942  023114  001002                        BNE     7$              ;BRANCH IF UNDERFLOW.
 4943
 4944                                                                ;REPORT BAD FEC ON EXPECTED OVERFLOW.
 4945  023116  104137              6$:     ERROR   137
 4946  023120  000771                        BR      4$
 4947
 4948  023122                      7$:                             ;REPORT BAD FEC ON EXPECTED UNDERFLOW.
 4949  023122  104140              8$:     ERROR   140
 4950  023124  000767                        BR      4$
 4951
 4952                                  ;REPORT INCORRECT FPS.
 4953  023126  005761  000026        10$:    TST     26(R1)          ;WAS THE RESULT OVER OR UNDER FLOW?
 4954  023132  001002                        BNE     12$             ;BRANCH IF UNDERFLOW.
 4955
 4956                                                                ;REPORT FPS BAD AFTER OVERFLOW.
 4957  023134  104141              11$:    ERROR   141
 4958  023136  000762                        BR      4$
 4959
 4960  023140                      12$:                            ;REPORT FPS BAD AFTER UNDERFLOW.
 4961  023140  104142              13$:    ERROR   142
 4962  023142  000760                        BR      4$
 4963
 4964                                  ;RESULT INCORRECT.
 4965  023144  012700  023262        15$:    MOV     #OVFTT,R0       ;SEE IF FAILURE IS ANTICIPATED
 4966  023150  010102                        MOV     R1,R2           ;FAILURE.
 4967  023152  062702  000014                ADD     #14,R2
 4968  023156  012703  000002                MOV     #2,R3
 4969  023162  022022              16$:    CMP     (R0)+,(R2)+
 4970  023164  001007                        BNE     17$             ;BRANCH IF NOT ANTICIPATED.
 4971  023166  077303                        SOB     R3,16$
 4972
 4973  023170  010102                        MOV     R1,R2           ;ERROR WAS ANTICIPATED SO RETURN
 4974  023172  062702  000034                ADD     #34,R2          ;TO THE ERROR REPORT IN THE CALLING
```

```
 4975  023176  010237  001236                MOV     R2,@#$TMP2      ;ROUTINE.
 4976  023202  000112                        JMP     (R2)
 4977
 4978  023204  005761  000026        17$:    TST     26(R1)          ;RESULT WAS NOT ANTICIPATED
 4979                                                                  ;SO ERROR MUST BE REPORTED HERE.
 4980                                                                  ;FIRST SEE IF ARGUMENTS SHOULD
 4981                                                                  ;HAVE RESULTED IN OVERFLOW OR UNDER
 4982                                                                  ;FLOW BY LOOKING AT THE FLAG.
 4983  023210  001002                        BNE     19$             ;BRANCH IF UNDERFLOW EXPECTED.
 4984
 4985                                                                  ;REPORT RESULT INCORRECT, EXPECTING
 4986  023212  104143                18$:    ERROR   143             ;OVERFLOW.
 4987  023214  000733                        BR      4$
 4988
 4989  023216                        19$:                            ;REPORT RESULT INCORRECT, EXPECTING
 4990  023216  104144                20$:    ERROR   144             ;UNDERFLOW.
 4991  023220  000731                        BR      4$
 4992
 4993                                ;IF NO FP TRAP OCCURS COME HERE.
 4994  023222  170204                25$:    STFPS   R4              ;GET FPS.
 4995  023224  170305                        STST    R5              ;GET FEC.
 4996  023226  012700  000200               MOV     #200,R0         ;SET FD MODE.
 4997  023232  170100                        LDFPS   R0
 4998  023234  012700  023262               MOV     #OVFTT,R0       ;GET THE RESULT.
 4999  023240  174010                        STD     AC0,(R0)
 5000  023242  010437  001250               MOV     R4,@#$TMP7
 5001  023246  010537  001254               MOV     R5,@#$TMP11
 5002  023252  010102                        MOV     R1,R2           ;ERROR WAS ANTICIPATED SO
 5003  023254  062702  000030               ADD     #30,R2          ;RETURN TO THE ERROR REPORT OF THE
 5004                                                                  ;CALLING ROUTINE.
 5005  023260  000112                        JMP     (R2)
 5006
 5007  023262  000000  000000  000000  OVFTT:  .WORD   0,0,0,0
 5008  023270  000000
 5009
 5010  023272                        KKKDONE:
 5011  023272  104412                        RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
 5012                                                                  ;SEE IF THE USER HAS EXPRESSED
 5013                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
 5014                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
 5015                                                                  ;THE USER TYPED CONTROL G?).
 5016
 5017
 5018
 5019
 5020                                ;;*****************************************************************
 5021                                ;*TEST 15       UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST
 5022                                ;*
 5023                                ;*THIS IS A TEST OF THE OVER FLOW AND UNDER FLOW CONDITIONS USING THE
 5024                                ;*MULD INSTRUCTION WITH TRAPS ENABLED. A SUBROUTINE IS USED TO SET UP
 5025                                ;*THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.
 5026                                ;*
 5027                                ;;*****************************************************************
 5028  023274  000004                TST15:  SCOPE
 5029
 5030                                ;UNDERFLOW, EXPONENT OF RESULT = -129
```

```
5031  023276                                  LLL1:
5032  023276  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5033  023300  004737  023622                          JSR     PC,@#OVUNDT
5034  023304  020052  125252                  1$:     .WORD   20052,125252    ;AC
5035  023310  125252  125252                          .WORD   125252,125252
5036  023314  020300  000000  000000        2$:     .WORD   20300,0,0,0     ;FSRC
5037  023322  000000
5038  023324  000177  177777  177777        3$:     .WORD   177,-1,-1,-1    ;RES
5039  023332  177777
5040  023334  000177  177777                4$:     .WORD   177,-1          ;ERROR RES.
5041  023340  125252  125252                          .WORD   125252,125252
5042  023344  002200                        5$:     2200                    ;FPS BEFORE EXECUTION.
5043  023346  102204                                102204                  ;FPS AFTER EXECUTION.
5044  023350  000012                        6$:     12                      ;FEC
5045  023352  177777                                -1                      ;FLAG
5046  023354  104157                        7$:     ERROR   157             ;ST 331 (BUT FIU) NO TRAP.
5047  023356  000401                                BR      8$
5048  023360  104160                                ERROR   160             ;ST 155 (BUT FD)
5049  023362                                8$:
5050
5051                                          ;UNDERFLOW, EXPONENT OF THE RESULT = -193
5052  023362                                  LLL2:
5053  023362  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5054  023364  004737  023622                          JSR     PC,@#OVUNDT
5055  023370  010327  127272                  1$:     .WORD   10327,127272    ;AC
5056  023374  036363  045454                          .WORD   36363,45454
5057  023400  010000  000000  000000        2$:     .WORD   10000,0,0,0     ;FSRC
5058  023406  000000
5059  023410  060127  127272                3$:     .WORD   60127,127272    ;RES
5060  023414  036363  045454                          .WORD   36363,45454
5061  023420  177777  177777  177777        4$:     .WORD   -1,-1,-1,-1     ;ERROR RES.
5062  023426  177777
5063  023430  007217                        5$:     7217                    ;FPS BEFORE EXECUTION.
5064  023432  107200                                107200                  ;FPS AFTER EXECUTION.
5065  023434  000012                        6$:     12                      ;FEC
5066  023436  177777                                -1                      ;FLAG
5067  023440  104161                        7$:     ERROR   161             ;ST 137 (BUT FIU) NO TRAP.
5068  023442  000401                                BR      8$
5069  023444  104156                                ERROR   156
5070  023446                                9$:
5071
5072                                          ;OVERFLOW, EXPONENT OF THE RESULT = 128
5073  023446                                  LLL3:
5074  023446  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5075  023450  004737  023622                          JSR     PC,@#OVUNDT
5076  023454  060252  125252                  1$:     .WORD   60252,125252    ;AC
5077  023460  125252  125252                          .WORD   125252,125252   ;FSRC
5078  023464  160100  000000  000000        2$:     .WORD   160100,0,0,0    ;FSRC
5079  023472  000000
5080  023474  100177  177777  177777        3$:     .WORD   100177,-1,-1,-1 ;RES
5081  023502  177777
5082  023504  100177  177777                4$:     .WORD   100177,-1       ;ERROR RES.
5083  023510  125252  125252                          .WORD   125252,125252
5084  023514  001200                        5$:     1200                    ;FPS BEFORE EXECUTION.
5085  023516  101216                                101216                  ;FPS AFTER EXECUTION.
5086  023520  000010                        6$:     10                      ;FEC
```

```
5087  023522  000000                        0                       ;FLAG
5088  023524  104162              7$:       ERROR   162             ;ST 333 (BUT FIV) NO TRAP.
5089  023526  000401                        BR      8$
5090  023530  104163                        ERROR   163             ;ST 700 (BUT FD).
5091  023532                      9$:
5092
5093                              ;OVERFLOW, EXPONENT OF THE RESULT = 130
5094  023532                      LLL4:
5095  023532  104412                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5096  023534  004737  023622                JSR     PC,@#OVUNDT
5097  023540  060345  067654      1$:       .WORD   60345,67654     ;AC
5098  023544  056765  045676                .WORD   56765,45676
5099  023550  060200  000000  000000  2$:   .WORD   60200,0,0,0     ;FSRC
5100  023556  000000
5101  023560  000345  067654      3$:       .WORD   345,67654       ;RES
5102  023564  056765  045676                .WORD   56765,45676
5103  023570  177777  177777  177777  4$:   .WORD   -1,-1,-1,-1     ;ERROR RES.
5104  023576  177777
5105  023600  007215              5$:       7215                    ;FPS BEFORE EXECUTION.
5106  023602  107202                        107202                  ;FPS AFTER EXECUTION.
5107  023604  000010              6$:       10                      ;FEC
5108  023606  000000                        0                       ;FLAG
5109  023610  104164              7$:       ERROR   164             ;ST 133 (BUT FIV) NO TRAP
5110  023612  000401                        BR      8$
5111  023614  104155                        ERROR   155
5112  023616  000137  024234      8$:       JMP     @#LLLDONE
5113
5114                              ;THIS SUBROUTINE, OVUNDT, IS USED TO SET UP THE OPERANDS, EXECUTE
5115                              ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
5116                              ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
5117                              ;TO IT IS MADE THUS:
5118                              ;
5119                              ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
5120                              ;               FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
5121                              ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
5122                              ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
5123                              ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
5124                              ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
5125                              ;               FEC:    .WORD   X               ;EXPECTED FEC
5126                              ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
5127                              ;               ERR1:   ERROR   X               ;TRAP ERROR.
5128                              ;               BR      CONT
5129                              ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
5130                              ;               CONT:                           ;RETURN ADDRESS
5131                              ;
5132                              ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
5133                              ;THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
5134                              ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
5135                              ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
5136                              ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
5137                              ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
5138                              ;IN THE SAME WAY  IF THE RESULT OF THE
5139                              ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
5140                              ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
5141                              ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
5142                              ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
```

```
                                        ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND CONDOT 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).

023622  012601              OVUNDT: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
023624  012700  000200              MOV     #200,R0         ;SET FD MODE.
023630  170100                      LDFPS   R0

023632  010100                      MOV     R1,R0           ;LOAD AC0, OPERAND.
023634  172410                      LDD     (R0),AC0

023636  010102                      MOV     R1,R2           ;SAVE THE DATA PATTERNS IN CASE OF
023640  010237  001240              MOV     R2,@#$TMP3      ;ERROR.
023644  062702  000010              ADD     #10,R2
023650  010237  001242              MOV     R2,@#$TMP4
023654  062702  000010              ADD     #10,R2
023660  010237  001244              MOV     R2,@#$TMP5
023664  016137  000042  001252      MOV     42(R1),@#$TMP10
023672  012737  024224  001246      MOV     #OVDTT,@#$TMP6

023700  016100  000040              MOV     40(R1),R0       ;LOAD THE FPS.
023704  170100                      LDFPS   R0
023706  012737  023730  001236      MOV     #1$,@#$TMP2
023714  012737  023740  000244      MOV     #50$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
                                                            ;OF ERROR.
023722  010100                      MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
023724  062700  000010              ADD     #10,R0

023730  171010              1$:     MULD    (R0),AC0        ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
023732  170000              2$:     CFCC

023734  000137  024164              JMP     @#25$           ;FAILURE, NO TRAP.

023740  011602              50$:    MOV     (SP),R2         ;TRAP TO HERE AND SEE IF THE PC OF THE
023742  020227  023732              CMP     R2,#2$          ;TRAP WAS THAT OF THE MULF INSTRUCTION.
023746  001402                      BEQ     51$             ;BRANCH IF YES.
023750  000137  036614              JMP     @#FPSPUR        ;OTHERWISE REPORT SPURIOUS FP ERROR.

023754  022626              51$:    CMP     (SP)+,(SP)+     ;RESET THE STACK
023756  170204                      STFPS   R4              ;GET FPS.
023760  170305                      STST    R5              ;GET FEC.
023762  012700  000200              MOV     #200,R0         ;SET FD MODE.
023766  170100                      LDFPS   R0
023770  012700  024224              MOV     #OVDTT,R0       ;GET THE RESULT.
023774  174010                      STD     AC0,(R0)
023776  010437  001250              MOV     R4,@#$TMP7
024002  010537  001254              MOV     R5,@#$TMP11

024006  012700  024224              MOV     #OVDTT,R0       ;CHECK THE RESULT.
024012  010102                      MOV     R1,R2
024014  062702  000020              ADD     #20,R2
024020  012703  000004              MOV     #4,R3
024024  022022              3$:     CMP     (R0)+,(R2)+
```

```
5199  024026  001027                      BNE     15$             ;BRANCH IF INCORRECT.
5200  024030  077303                      SOB     R3,3$
5201
5202  024032  026104  000042              CMP     42(R1),R4       ;WAS FPS CORRECT?
5203  024036  001014                      BNE     10$             ;BRANCH IF FPS IS INCORRECT.
5204
5205  024040  026105  000044              CMP     44(R1),R5       ;IS FEC CORRECT?
5206  024044  001002                      BNE     5$              ;IF INCORRECT BRANCH.
5207  024046  000161  000056      4$:     JMP     56(R1)          ;RETURN, TEST COMPLETED.
5208
5209                                      ;REPORT INCORRECT FEC.
5210  024052  005761  000046      5$:     TST     46(R1)          ;WAS THE RESULT OVERFLOW OR UNDERFLOW?
5211  024056  001002                      BNE     7$              ;BRANCH IF UNDERFLOW.
5212
5213                                                              ;REPORT BAD FEC ON EXPECTED OVERFLOW.
5214  024060  104151              6$:     ERROR   151
5215  024062  000771                      BR      4$
5216
5217  024064                      7$:                             ;REPORT BAD FEC ON EXPECTED UNDERFLOW.
5218  024064  104152              8$:     ERROR   152
5219  024066  000767                      BR      4$
5220
5221                                      ;REPORT INCORRECT FPS.
5222  024070  005761  000046      10$:    TST     46(R1)          ;WAS THE RESULT OVER OR UNDER FLOW?
5223  024074  001002                      BNE     12$             ;BRANCH IF UNDERFLOW.
5224
5225                                                              ;REPORT FPS BAD AFTER OVERFLOW.
5226  024076  104153              11$:    ERROR   153
5227  024100  000762                      BR      4$
5228
5229  024102                      12$:                            ;REPORT FPS BAD AFTER UNDERFLOW.
5230  024102  104154              13$:    ERROR   154
5231  024104  000760                      BR      4$
5232
5233                                      ;RESULT INCORRECT.
5234  024106  012700  024224      15$:    MOV     #OVDTT,R0       ;SEE IF FAILURE IS ANTICIPATED
5235  024112  010102                      MOV     R1,R2           ;FAILURE.
5236  024114  062702  000030              ADD     #30,R2
5237  024120  012703  000004              MOV     #4,R3
5238  024124  022022              16$:    CMP     (R0)+,(R2)+
5239  024126  001007                      BNE     17$             ;BRANCH IF NOT ANTICIPATED.
5240  024130  077303                      SOB     R3,16$
5241
5242  024132  010102                      MOV     R1,R2           ;ERROR WAS ANTICIPATED SO RETURN
5243  024134  062702  000054              ADD     #54,R2          ;TO THE ERROR REPORT IN THE CALLING
5244  024140  010237  001236              MOV     R2,@#$TMP2      ;ROUTINE.
5245  024144  000112                      JMP     (R2)
5246
5247  024146  005761  000046      17$:    TST     46(R1)          ;RESULT WAS NOT ANTICIPATED
5248                                                              ;SO ERROR MUST BE REPORTED HERE.
5249                                                              ;FIRST SEE IF ARGUMENTS SHOULD
5250                                                              ;HAVE RESULTED IN OVERFLOW OR UNDER
5251                                                              ;FLOW BY LOOKING AT THE FLAG.
5252  024152  001002                      BNE     19$             ;BRANCH IF UNDERFLOW EXPECTED.
5253
5254                                                              ;REPORT RESULT INCORRECT, EXPECTING
```

```
5255  024154  104155                  18$:    ERROR   155             ;OVERFLOW.
5256  024156  000733                          BR      4$
5257
5258  024160                          19$:                            ;REPORT RESULT INCORRECT, EXPECTING
5259  024160  104156                  20$:    ERROR   156             ;UNDERFLOW.
5260  024162  000731                          BR      4$
5261
5262                                  ;IF NO FP TRAP OCCURS COME HERE.
5263  024164  170204                  25$:    STFPS   R4              ;GET FPS.
5264  024166  170305                          STST    R5              ;GET FEC.
5265  024170  012700  000200                  MOV     #200,R0         ;SET FD MODE.
5266  024174  170100                          LDFPS   R0
5267  024176  012700  024224                  MOV     #OVDTT,R0       ;GET THE RESULT.
5268  024202  174010                          STD     AC0,(R0)
5269  024204  010437  001250                  MOV     R4,@#$TMP7
5270  024210  010537  001254                  MOV     R5,@#$TMP11
5271  024214  010102                          MOV     R1,R2           ;ERROR WAS ANTICIPATED SO
5272  024216  062702  000050                  ADD     #50,R2          ;RETURN TO THE ERROR REPORT OF THE
5273                                                                  ;CALLING ROUTINE.
5274  024222  000112                          JMP     (R2)
5275
5276  024224  003000  000000  000000  OVDTT:  .WORD   0,0,0,0
5277  024232  000000
5278
5279  024234                          LL.DONE:
5280  024234  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
5281                                                                  ;SEE IF THE USER HAS EXPRESSED
5282                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
5283                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5284                                                                  ;THE USER TYPED CONTROL G?).
5285
5286
5287
5288
5289
5290                                  ;;*****************************************************************
5291                                  ;*TEST 16       MODF TEST
5292                                  ;*
5293                                  ;*THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF
5294                                  ;*A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
5295                                  ;*AND CHECK THE RESULTS.
5296                                  ;*
5297                                  ;*****************************************************************
5298  024236  000004                  TST16:  SCOPE
5299
5300                                  ;MODF WITH (FSRC=AC=0)
5301  024240                          GGG1:
5302  024240  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5303  024242  004737  025324                  JSR     PC,@#MODFSUB
5304  024246  000000  000000          1$:     .WORD   0,0             ;AC
5305  024252  000000  000000          2$:     .WORD   0,0             ;FSRC
5306  024256  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
5307  024262  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
5308  024266  177777  177777          5$:     .WORD   -1,-1           ;ERROR FRACTIONAL RES.
5309  024272  177777  177777          6$:     .WORD   -1,-1           ;ERROR INGETER RES.
5310  024276  000013                  7$:     13                      ;FPS BEFORE EXECUTION.
```

```
5311  024300  000004                        4                       ;FPS AFTER EXECUTION.
5312  024302  104056                8$:     ERROR   56              ;STORE SINGLE ZERO BAD.
5313  024304  000401                        BR      9$
5314  024306  104057                        ERROR   57              ;AC V 1 <= ZERO FAILED.
5315  024310                        9$:
5316
5317                                ;MODF TEST, WITH (FSRC=0)
5318  024310                        GGG2:
5319  024310  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5320  024312  004737  025324                JSR     PC,@#MODFSUB
5321  024316  123456  076543        1$:     .WORD   123456,76543    ;AC
5322  024322  000000  000000        2$:     .WORD   0,0             ;FSRC
5323  024326  000000  000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
5324  024332  000000  000000        4$:     .WORD   0,0             ;INTEGER RESULT.
5325  024336  123456  076543        5$:     .WORD   123456,76543    ;ERROR FRACTIONAL RES.
5326  024342  177777  177777        6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5327  024346  000000                7$:     0                       ;FPS BEFORE EXECUTION.
5328  024350  000004                        4                       ;FPS AFTER EXECUTION.
5329  024352  104056                8$:     ERROR   56              ;STORE ZERO FAILURE.
5330  024354  000401                        BR      9$
5331  024356  104057                        ERROR   57
5332  024360                        9$:
5333
5334                                ;MODF TEST WITH (AC=0)
5335  024360                        GGG3:
5336  024360  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5337  024362  004737  025324                JSR     PC,@#MODFSUB
5338  024366  000000  000000        1$:     .WORD   0,0             ;AC
5339  024372  076543  021234        2$:     .WORD   76543,21234     ;FSRC
5340  024376  000000  000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
5341  024402  000000  000000        4$:     .WORD   0,0             ;INTEGER RES.
5342  024406  000000  000000        5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
5343  024412  177777  177777        6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5344  024416  000003                7$:     3                       ;FPS BEFORE EXECUTION.
5345  024420  000004                        4                       ;FPS AFTER EXECUTION.
5346  024422  104053                8$:     ERROR   53              ;RES.BAD
5347  024424  000401                        BR      9$
5348  024426  104057                        ERROR   57
5349  024430                        9$:
5350
5351                                ;MODF TEST WITH EXPONENT OF THE RESULT = 25
5352  024430                        GGG4:
5353  024430  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5354  024432  004737  025324                JSR     PC,@#MODFSUB
5355  024436  046252  125252        1$:     .WORD   46252,125252    ;AC
5356  024442  040300  000000        2$:     .WORD   40300,0         ;FSRC
5357  024446  000000  000000        3$:     .WORD   0,0             ;FRACTIONAL RES.
5358  024452  046377  177777        4$:     .WORD   46377,-1        ;INTEGER RES.
5359  024456  046252  125252        5$:     .WORD   46252,125252    ;ERROR FRACTIONAL RES.
5360  024462  040300  000000        6$:     .WORD   40300,0         ;ERROR INTEGER RES.
5361  024466  000013                7$:     13                      ;FPS BEFORE EXECUTION.
5362  024470  000004                        4                       ;FPS AFTER EXECUTION.
5363  024472  104053                8$:     ERROR   53              ;ST 134
5364  024474  000401                        BR      9$
5365  024476  104060                        ERROR   60
5366  024500                        9$:
```

```
5367
5368                                        ;MODF TEST WITH EXPONENT OF THE RESULT = 127
5369  024500                                GGG5:
5370  024500  104413                              LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS
5371  024502  004737  025324                      JSR     PC,@#MODFSUB
5372  024506  077652  125252              1$:     .WORD   77652,125252    ;AC
5373  024512  040300  000000              2$:     .WORD   40300,0         ;FSRC
5374  024516  000000  000000              3$:     .WORD   0,0             ;FRACTIONAL RES.
5375  024522  077777  177777              4$:     .WORD   77777,-1        ;INTEGER RES.
5376  024526  077652  125252              5$:     .WORD   77652,125252    ;ERROR FRACTIONAL RES.
5377  024532  040300  000000              6$:     .WORD   40300,0         ;ERROR INTEGER RES.
5378  024536  000000                      7$:     0                       ;FPS BEFORE EXECUTION.
5379  024540  000004                              4                       ;FPS AFTER EXECUTION.
5380  024542  104053                      8$:     ERROR   53
5381  024544  000401                              BR      9$
5382  024546  104060                              ERROR   60
5383  024550                              9$:
5384
5385                                        ;MODF TEST WITH EXPONENT OF RESULT = 25
5386  024550                                GGG6:
5387  024550  104413                              LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5388  024552  004737  025324                      JSR     PC,@#MODFSUB
5389  024556  046200  000001              1$:     .WORD   46200,1         ;AC
5390  024562  040340  000000              2$:     .WORD   40340,0         ;FSRC
5391  024566  000000  000000              3$:     .WORD   0,0             ;FRACTIONAL RES.
5392  024572  046340  000001              4$:     .WORD   46340,1         ;INTEGER RES.
5393  024576  040000  000000              5$:     .WORD   40000,0         ;ERROR FRACTIONAL RES.
5394  024602  177777  177777              6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5395  024606  000013                      7$:     13                      ;FPS BEFORE EXECUTION.
5396  024610  000004                              4                       ;FPS AFTER EXECUTION.
5397  024612  104061                      8$:     ERROR   61              ;BAD CONSTANT (NOT 24),
5398                                              ;OR ST 525 TO 050 INTO 150.
5399  024614  000401                              BR      9$
5400  024616  104054                              ERROR   54
5401  024620                              9$:
5402
5403                                        ;MODF   TEST WITH EXPONENT OF THE RESULT = 24
5404  024620                                GGG7:
5405  024620  104413                              LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5406  024622  004737  025324                      JSR     PC,@#MODFSUB
5407  024626  046000  000001              1$:     .WORD   46000,1         ;AC
5408  024632  040340  000000              2$:     .WORD   40340,0         ;FSRC
5409  024636  040100  000000              3$:     .WORD   40100,0         ;FRACTIONAL RES.
5410  024642  046140  000001              4$:     .WORD   46140,1         ;INTEGER RESULT.
5411  024646  000000  000000              5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
5412  024652  177777  177777              6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5413  024656  000000                      7$:     0                       ;FPS BEFORE EXECUTION.
5414  024660  000000                              0                       ;FPS AFTER EXECUTION.
5415  024662  104062                      8$:     ERROR   62              ;BAD CONSTANT USED (NOT 24)
5416                                              ;OR ST 525 TO 150 INTO 050
5417  024664  000401                              BR      9$
5418  024666  104054                              ERROR   54
5419  024670                              9$:
5420
5421                                        ;MODF TEST WITH EXPONENT OF THE RESULT = 10
5422  024670                                GGG8:
```

```
5423  024670  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5424  024672  004737  025324                  JSR     PC,@#MODFSUB
5425  024676  042577  177777          1$:     .WORD   42577,-1        ;AC
5426  024702  040200  000000          2$:     .WORD   40200,0         ;FSRC
5427  024706  040177  176000          3$:     .WORD   40177,176000    ;FRACTIONAL RES.
5428  024712  042577  140000          4$:     .WORD   42577,140000    ;INTEGER RES.
5429  024716  177777  177777          5$:     .WORD   -1,-1           ;ERROR FRACTIONAL RES.
5430  024722  177777  177777          6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5431  024726  000000                  7$:     0                       ;FPS BEFORE EXECUTION.
5432  024730  000000                          0                       ;FPS AFTER EXECUTION.
5433  024732  104053                  8$:     ERROR   53
5434  024734  000401                          BR      9$
5435  024736  104054                          ERROR   54
5436  024740                          9$:
5437
5438                                  ;MODF TEST WITH THE EXPONENT OF THE RESULT = 10
5439  024740                          GGG9:
5440  024740  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5441  024742  004737  025324                  JSR     PC,@#MODFSUB
5442  024746  042577  140001          1$:     .WORD   42577,140001    ;AC
5443  024752  040200  000000          2$:     .WORD   40200,0         ;FSRC
5444  024756  034600  000000          3$:     .WORD   34600,0         ;FRACTIONAL RES.
5445  024762  042577  140000          4$:     .WORD   42577,140000    ;INTEGER RES.
5446  024766  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
5447  024772  177777  177777          6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5448  024776  000000                  7$:     0                       ;FPS BEFORE EXECUTION.
5449  025000  000000                          0                       ;FPS AFTER EXECUTION.
5450  025002  104063                  8$:     ERROR   63              ;ST 532 TO 122 INTO NORMALIZE.
5451  025004  000401                          BR      9$
5452  025006  104054                          ERROR   54
5453  025010                          9$:
5454
5455                                  ;MODF TEST WITH EXPONENT OF THE RESULT = 9
5456  025010                          GGG10:
5457  025010  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5458  025012  004737  025324                  JSR     PC,@#MODFSUB
5459  025016  042377  100000          1$:     .WORD   42377,100000    ;AC
5460  025022  040200  000000          2$:     .WORD   40200,0         ;FSRC
5461  025026  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
5462  025032  042377  100000          4$:     .WORD   42377,100000    ;INTEGER RES.
5463  025036  177777  177777          5$:     .WORD   -1,-1           ;ERROR FRACTIONAL RES.
5464  025042  177777  177777          6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
5465  025046  000013                  7$:     13                      ;FPS BEFORE EXECUTION.
5466  025050  000004                          4                       ;FPS AFTER EXECUTION.
5467  025052  104053                  8$:     ERROR   53
5468  025054  000401                          BR      9$
5469  025056  104054                          ERROR   54
5470  025060                          9$:
5471
5472                                  ;MODF TEST WITH EXPONENT OF THE RESULT = 0
5473  025060                          GGG11:
5474  025060  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5475  025062  004737  025324                  JSR     PC,@#MODFSUB
5476  025066  040177  177777          1$:     .WORD   40177,-1        ;AC
5477  025072  040200  000000          2$:     .WORD   40200,0         ;FSRC
5478  025076  040177  177777          3$:     .WORD   40177,-1        ;FRACTIONAL RES.
```

```
5479  025102  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
5480  025106  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
5481  025112  040177  177777          6$:     .WORD   40177,-1        ;ERROR INTEGER RES.
5482  025116  000017                  7$:     17                      ;FPS BEFORE EXECUTION.
5483  025120  000000                          0                       ;FPS AFTER EXECUTION.
5484  025122  104064                  8$:     ERROR   64              ;ST 041 TO 046 INTO 246.
5485  025124  000401                          BR      9$
5486  025126  104064                          ERROR   64
5487  025130                          9$:
5488
5489                                  ;MODF TEST WITH EXPONENT OF THE RESULT = -15
5490  025130                          GGG12:
5491  025130  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5492  025132  004737  025324                  JSR     PC,@#MODFSUB
5493  025136  034377  177777          1$:     .WORD   34377,-1        ;AC
5494  025142  040200  000000          2$:     .WORD   40200,0         ;FSRC
5495  025146  034377  177777          3$:     .WORD   34377,-1        ;FRACTIONAL RES.
5496  025152  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
5497  025156  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
5498  025162  034377  177777          6$:     .WORD   34377,-1        ;ERROR INTEGER RES.
5499  025166  000000                  7$:     0                       ;FPS BEFORE EXECUTION.
5500  025170  000000                          0                       ;FPS AFTER EXECUTION.
5501  025172  104064                  8$:     ERROR   64
5502  025174  000401                          BR      9$
5503  025176  104064                          ERROR   64
5504  025200                          9$:
5505
5506                                  ;MODF TEST WITH EXPONENT OF RESULT = -64, IN ROUND MODE
5507  025200                          GGG13:
5508  025200  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5509  025202  004737  025324                  JSR     PC,@#MODFSUB
5510  025206  020000  000001          1$:     .WORD   20000,1         ;AC
5511  025212  040300  000000          2$:     .WORD   40300,0         ;FSRC
5512  025216  020100  000002          3$:     .WORD   20100,2         ;FRACTIONAL RES.
5513  025222  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
5514  025226  020100  000001          5$:     .WORD   20100,1         ;ERROR FRACTIONAL RES.
5515  025232  000000  000000          6$:     .WORD   0,0             ;ERROR INTEGER RES.
5516  025236  000000                  7$:     0                       ;FPS BEFORE EXECUTION.
5517  025240  000000                          0                       ;FPS AFTER EXECUTION.
5518  025242  104065                  8$:     ERROR   65              ;ROUND TRUNK, ST 126 INTO ROUND.
5519  025244  000401                          BR      9$
5520  025246  104054                          ERROR   54
5521  025250                          9$:
5522
5523                                  ;MODF TEST WITH EXPONENT OF RESULT = 11
5524  025250                          GGG14:
5525  025250  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5526  025252  004737  025324                  JSR     PC,@#MODFSUB
5527  025256  142777  170000          1$:     .WORD   142777,170000   ;AC
5528  025262  040200  000000          2$:     .WORD   40200,0         ;FSRC
5529  025266  140000  000000          3$:     .WORD   140000,0        ;FRACTIONAL RES.
5530  025272  142777  160000          4$:     .WORD   142777,160000   ;INTEGER RES.
5531  025276  040000  000000          5$:     .WORD   40000,0         ;ERROR FRACTIONAL RES.
5532  025302  042777  160000          6$:     .WORD   42777,160000    ;ERROR INTEGER RES.
5533  025306  000007                  7$:     7                       ;FPS BEFORE EXECUTION.
5534  025310  000010                          10                      ;FPS AFTER EXECUTION.
```

```
005535  025312  104066                  8$:     ERROR   66              ;SIGN OF FRACTION.
005536  025314  000401                          BR      9$
005537  025316  104067                          ERROR   67              ;SIGN OF INTEGER.
005538  025320  000167  000366          9$:     JMP     GGGDONE         ;GO TO NEXT TEST.
005539
005540
005541                                  ;THIS SUBROUTINE, MODFSUB, IS CALLED TO SETUP THE
005542                                  ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
005543                                  ;IT IS CALLED THUS:
005544                                  ;
005545                                  ;               ACARG:   .WORD   X,X             ;AC OPERAND
005546                                  ;               FSRCARG:.WORD    X,X             ;FSRC OPERAND
005547                                  ;               FRES:    .WORD   X,X             ;FRACTIONAL RESULT
005548                                  ;               INTRES:  .WORD   X,X             ;INTEGER RESULT
005549                                  ;               ERFRES:  .WORD   X,X             ;ERROR FRACTION RESULT
005550                                  ;               ERINTRES:.WORD   X,X             ;ERROR INTEGER RESULT
005551                                  ;               FPSB:    .WORD   X               ;FPS BEFORE EXECUTION
005552                                  ;               FPSA:    .WORD   X               ;FPS AFTER EXECUTION
005553                                  ;               ERR1:    ERROR   X               ;FRACTION ERROR
005554                                  ;                        BR      CONT
005555                                  ;               ERR2:    ERROR   X               ;INTEGEP ERROR
005556                                  ;               CONT:                            ;RETURN ADDRESS
005557                                  ;
005558                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
005559                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
005560                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
005561                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
005562                                  ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
005563                                  ;THEN MODFSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
005564                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
005565                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
005566                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
005567                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODFSUB PASSES CONTROL TO THE
005568                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
005569                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
005570                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
005571                                  ;IF A MATCH IS MADE HOWEVER, MODFSUB WILL RETURN CONTROL TO THE ERROR
005572                                  ;CALL AT ERR2.
005573
005574  025324  012601                  MODFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
005575  025326  012700  000200                  MOV     #200,R0         ;SET FD MODE.
005576  025332  170100                          LDFPS   R0
005577  025334  010100                          MOV     R1,R0           ;SET UP AC0
005578  025336  172410                          LDD     (R0),AC0
005579  025340  012700  025702                  MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
005580  025344  172510                          LDD     (R0),AC1
005581  025346  016100  000030                  MOV     30(R1),R0       ;SET UP THE FPS.
005582  025352  170100                          LDFPS   R0
005583  025354  012737  025370  001236          MOV     #1$,@#$TMP2
005584  025362  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
005585  025364  062700  000004                  ADD     #4,R0
005586
005587  025370  171410                  1$:     MODF    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
005588
005589  025372  170204                          STFPS   R4              ;GET THE FPS.
005590  025374  012700  000200                  MOV     #200,R0         ;SET FD MODE.
```

```
5591  025400  170100                        LDFPS   R0
5592  025402  012700  025662                MOV     #MODFT0,R0      ;GET THE FRACTIONAL RESULT.
5593  025406  174010                        STD     AC0,(R0)
5594  025410  012700  025672                MOV     #MODFT1,R0      ;GET THE INTEGER RESULT.
5595  025414  174110                        STD     AC1,(R0)
5596
5597  025416  010102                        MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
5598  025420  010237  001240                MOV     R2,@#$TMP3
5599  025424  062702  000004                ADD     #4,R2
5600  025430  010237  001242                MOV     R2,@#$TMP4
5601  025434  062702  000004                ADD     #4,R2
5602  025440  010237  001244                MOV     R2,@#$TMP5
5603  025444  062702  000004                ADD     #4,R2
5604  025450  010237  001246                MOV     R2,@#$TMP6
5605  025454  012737  025662  001250        MOV     #MODFT0,@#$TMP7
5606  025462  012737  025672  001252        MOV     #MODFT1,@#$TMP10
5607  025470  010437  001254                MOV     R4,@#$TMP11
5608  025474  016137  000032  001256        MOV     32(R1),@#$TMP12
5609
5610  025502  012702  025662                MOV     #MODFT0,R2      ;CHECK THE FRACTIONAL RESULT.
5611  025506  026112  000010                CMP     10(R1),(R2)
5612  025512  001022                        BNE     10$             ;BRANCH IF INCORRECT.
5613  025514  026162  000012  000002        CMP     12(R1),2(R2)
5614  025522  001016                        BNE     10$
5615
5616  025524  012702  025672                MOV     #MODFT1,R2      ;CHECK THE INTEGER RESULT.
5617  025530  026112  000014                CMP     14(R1),(R2)
5618  025534  001026                        BNE     15$             ;BRANCH IF INCORRECT.
5619  025536  026162  000016  000002        CMP     16(R1),2(R2)
5620  025544  001022                        BNE     15$
5621
5622  025546  026104  000032                CMP     32(R1),R4       ;CHECK THE FPS.
5623  025552  001034                        BNE     20$             ;BRANCH IF INCORRECT.
5624
5625  025554  000161  000042        9$:     JMP     42(R1)          ;RETURN.
5626
5627                                ;FRACTIONAL ERROR.
5628  025560  026112  000020        10$:    CMP     20(R1),(R2)     ;WAS THE ERROR ANTICIPATED?
5629  025564  001010                        BNE     11$             ;BRANCH IF NOT ANTICIPATED.
5630  025566  026162  000022  000002        CMP     22(R1),2(R2)
5631  025574  001004                        BNE     11$
5632  025576  010102                        MOV     R1,R2           ;THE ERROR WAS ANTICIPATED SO
5633  025600  062702  000034                ADD     #34,R2          ;RETURN TO THE ERROR REPORT AT THE
5634                                                                ;CALLING ROUTINE.
5635  025604  000112                        JMP     (R2)
5636
5637  025606                        11$:                            ;THE ERROR WAS NOT ANTICIPATED SO
5638  025606  104053                12$:    ERROR   53              ;REPORT THE INCORRECT FRACTION HERE.
5639  025610  000761                        BR      9$
5640
5641                                ;INTEGER ERROR.
5642  025612  026112  000024        15$:    CMP     24(R1),(R2)     ;WAS THIS ERROR ANTICIPATED?
5643  025616  001010                        BNE     16$             ;BRANCH IF NOT.
5644  025620  026162  000026  000002        CMP     26(R1),2(R2)
5645  025626  001004                        BNE     16$
5646  025630  010102                        MOV     R1,R2           ;THE ERROR WAS ANTICIAPTED SO RETURN
```

```
5647  025632  062702  000040              ADD     #40,R2          ;TO THE ERROR REPORT IN THE CALLING
5648                                                              ;ROUTINE.
5649  025636  000112                      JMP     (R2)
5650
5651  025640                      16$:                            ;THE ERROR WAS NOT ANTICIPATED SO REPORT
5652  025640  104054              17$:    ERROR   54              ;THE INTEGER FAILURE HERE.
5653  025642  000744                      BR      9$
5654
5655                              ;FPS INCORRECT.
5656  025644  010437  001254      20$:    MOV     R4,@#$TMP11     ;REPORT INCORRECT FPS.
5657  025650  016137  000032  001256      MOV     32(R1),@#$TMP12
5658  025656  104055              21$:    ERROR   55
5659  025660  000735                      BR      9$
5660
5661  025662  000000  000000  000000  MODFT0: .WORD   0,0,0,0
5662  025670  000000
5663
5664  025672  000000  000000  000000  MODFT1: .WORD   0,0,0,0
5665  025700  000000
5666
5667  025702  177777  177777  177777  MODP1:  .WORD   -1,-1,-1,-1
5668  025710  177777
5669
5670  025712                      GGGDONE:
5671  025712  104412                      RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
5672                                                              ;SEE IF THE USER HAS EXPRESSED
5673                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
5674                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5675                                                              ;THE USER TYPED CONTROL G?).
5676
5677
5678
5679
5680                              ;;*************************************************************
5681                              ;*TEST 17       MODD TEST
5682                              ;*
5683                              ;*THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE
5684                              ;*TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE
5685                              ;*RESULTS.
5686                              ;*
5687                              ;;*************************************************************
5688  025714  000004              TST17:  SCOPE
5689
5690                              ;MODD WITH (FSRC=AC=0)
5691  025716                      HHH1:
5692  025716  104413                      LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5693  025720  004737  027422              JSR     PC,@#MODDSUB
5694  025724  000000  000000  000000  1$:     .WORD   0,0,0,0         ;AC
5695  025732  000000
5696  025734  000000  000000  000000  2$:     .WORD   0,0,0,0         ;FSRC
5697  025742  000000
5698  025744  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
5699  025752  000000
5700  025754  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
5701  025762  000000
5702  025764  000000  000000  000000  5$:     .WORD   0,0,0,0         ;ERROR FRACTIONAL RES.
```

```
5703  025772  000000
5704  025774  000000  000000  177777  6$:     .WORD   0,0,-1,-1        ;ERROR INTEGER RES.
5705  026002  177777
5706  026004  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
5707  026006  000204                          204                     ;FPS AFTER EXECUTION.
5708  026010  104070                  8$:     ERROR   70
5709  026012  000401                          BR      9$
5710  026014  104074                          ERROR   74              ;ST 231 TO 142 INTO 143
5711  026016                          9$:
5712
5713                                  ;MODD TEST WITH FSRC=0
5714  026016                          HHH2:
5715  026016  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5716  026020  004737  027422                  JSR     PC,@#MODDSUB
5717  026024  012345  067012          1$:     .WORD   012345,67012    ;AC
5718  026030  034567  012345                  .WORD   34567,012345
5719  026034  000000  000000  000000  2$:     .WORD   0,0,0,0         ;FSRC
5720  026042  000000
5721  026044  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
5722  026052  000000
5723  026054  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
5724  026062  000000
5725  026064  012345  067012          5$:     .WORD   012345,67012    ;ERROR FRACTIONAL RES.
5726  026070  034567  012345                  .WORD   34567,012345
5727  026074  177777  177777  177777  6$:     .WORD   -1,-1,-1,-1     ;ERROR INTEGER RES.
5728  026102  177777
5729  026104  000213                  7$:     213                     ;FPS BEFORE EXECUTION.
5730  026106  000204                          204                     ;FPS AFTER EXECUTION.
5731  026110  104075                  8$:     ERROR   75              ;STORE DOUBLE ZERO
5732  026112  000401                          BR      9$
5733  026114  104076                          ERROR   76              ;AC V 1 <= ZERO ST 143
5734  026116                          9$:
5735
5736                                  ;MODD TEST WITH (AC=0)
5737  026116                          HHH3:
5738  026116  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5739  026120  004737  027422                  JSR     PC,@#MODDSUB
5740  026124  000000  000000  000000  1$:     .WORD   0,0,0,0         ;AC
5741  026132  000000
5742  026134  072727  127272          2$:     .WORD   72727,127272    ;FSRC
5743  026140  072727  127272                  .WORD   72727,127272
5744  026144  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
5745  026152  000000
5746  026154  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
5747  026162  000000
5748  026164  177777  177777  177777  5$:     .WORD   -1,-1,-1,-1     ;ERROR FRACTIONAL RES.
5749  026172  177777
5750  026174  177777  177777  177777  6$:     .WORD   -1,-1,-1,-1     ;ERROR INTEGER RES.
5751  026202  177777
5752  026204  000213                  7$:     213                     ;FPS BEFORE EXECUTION.
5753  026206  000204                          204                     ;FPS AFTER EXECUTION.
5754  026210  104070                  8$:     ERROR   70
5755  026212  000401                          BR      9$
5756  026214  104071                          ERROR   71
5757  026216                          9$:
5758
```

```
                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 57
026216                                  HHH4:
026216  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
026220  004737  027422                          JSR     PC,@#MODDSUB
026224  056252  125252                  1$:     .WORD   56252,125252    ;AC
026230  125252  125250                          .WORD   125252,125250
026234  040300  000000  000000          2$:     .WORD   40300,0,0,0     ;FSRC
026242  000000
026244  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
026252  000000
026254  056377  177777  177777          4$:     .WORD   56377,-1,-1,-4  ;INTEGER RES.
026262  177774
026264  000000  000000                  5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
026270  125252  125252                          .WORD   125252,125252
026274  056377  177777  177777          6$:     .WORD   56377,-1,-1,-1  ;ERROR INTEGER RES.
026302  177777
026304  000213                          7$:     213                     ;FPS BEFORE EXECUTION.
026306  000204                                  204                     ;FPS AFTER EXECUTION.
026310  104077                          8$:     ERROR   77              ;ST 526 TO 134 INTO 135
026312  000401                                  BR      9$
026314  104077                                  ERROR   77
026316                                  9$:

                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 79
026316                                  HHH5:
026316  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
026320  004737  027422                          JSR     PC,@#MODDSUB
026324  140240  000000  000000          1$:     .WORD   140240,0,0,0    ;AC
026332  000000
026334  063714  146314                  2$:     .WORD   63714,146314    ;FSRC
026340  133572  167737                          .WORD   133572,167737
026344  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
026352  000000
026354  163777  177777                  4$:     .WORD   163777,-1       ;INTEGER RES.
026360  162531  125726                          .WORD   162531,125726
026364  177777  177777  177777          5$:     .WORD   -1,-1,-1,-1     ;ERROR FRACTIONAL RES.
026372  177777
026374  063777  177777                  6$:     .WORD   63777,-1        ;ERROR INTEGER RES.
026400  162531  125726                          .WORD   162531,125726
026404  000210                          7$:     210                     ;FPS BEFORE EXECUTION.
026406  000204                                  204                     ;FPS AFTER EXECUTION.
026410  104070                          8$:     ERROR   70
026412  000401                                  BR      9$
026414  104100                                  ERROR   100             ;ST 526 BAD SIGN
026416                                  9$:

                                        ;MODD TEST WITH EXPONENT OF THE RESULT = 57
026416                                  HHH6:
026416  104413                                  LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
026420  004737  027422                          JSR     PC,@#MODDSUB
026424  056200  000000  000000          1$:     .WORD   56200,0,0,1     ;AC
026432  000001
026434  040340  000000  000000          2$:     .WORD   40340,0,0,0     ;FSRC
026442  000000
026444  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
026452  000000
```

```
5815  026454  056340  000000  000000  4$:     .WORD   56340,0,0,1      ;INTEGER RES.
5816  026462  000001
5817  026464  040000  000000  000000  5$:     .WORD   40000,0,0,0      ;ERROR FRACTIONAL RES.
5818  026472  000000
5819  026474  056340  000000  000000  6$:      WORD   56340,0,0,1      ;ERROR INTEGER RES.
5820  026502  000001
5821  026504  000213                  7$:     213                      ;FPS BEFORE EXECUTION.
5822  026506  000204                          204                      ;FPS AFTER EXECUTION.
5823  026510  104101                  8$:     ERROR   101              ;CONSTANT BAD (NOT 56)
5824                                                                   ;OR ST 525 TO 050 INTO 150
5825  026512  000401                          BR      9$
5826  026514  104101                          ERROR   101
5827  026516                          9$:
5828
5829                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 56
5830  026516                          MMM7:
5831  026516  104413                          LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
5832  026520  004737  027422                  JSR     PC,@#MODDSUB
5833  026524  056000  000000  000000  1$:     .WORD   56000,0,0,1      ;AC
5834  026532  000001
5835  026534  040340  000000  000000  2$:     .WORD   40340,0,0,0      ;FSRC
5836  026542  000000
5837  026544  040100  000000  000000  3$:     .WORD   40100,0,0,0      ;FRACTIONAL RES.
5838  026552  000000
5839  026554  056140  000000  000000  4$:     .WORD   56140,0,0,1      ;INTEGER RES.
5840  026562  000001
5841  026564  000000  000000  000000  5$:     .WORD   0,0,0,0          ;ERROR FRACTIONAL RES.
5842  026572  000000
5843  026574  056140  000000  000000  6$:     .WORD   56140,0,0,1      ;ERROR INTEGER RES.
5844  026602  000001
5845  026604  000213                  7$:     213                      ;FPS BEFORE EXECUTION.
5846  026606  000200                          200                      ;FPS AFTER EXECUTION.
5847  026610  104102                  8$:     ERROR   102              ;BAD CONSTANT (NOT 56) OR
5848                                                                   ;ST 525 TO 150 INTO 050
5849  026612  000401                          BR      9$
5850  026614  104102                          ERROR   102
5851  026616                          9$:
5852
5853                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 36
5854  026616                          MMM8:
5855  026616  104413                          LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
5856  026620  004737  027422                  JSR     PC,@#MODDSUB
5857  026624  051177  177777  177777  1$:     .WORD   51177,-1,-1,-1   ;AC
5858  026632  177777
5859  026634  040200  000000  000000  2$:     .WORD   40200,0,0,0      ;FSRC
5860  026642  000000
5861  026644  040177  177760  000000  3$:     .WORD   40177,-20,0,0    ;FRACTIONAL RES.
5862  026652  000000
5863  026654  051177  177777  177760  4$:     .WORD   51177,-1,-20,0   ;INTEGER RES.
5864  026662  000000
5865  026664  177777  177777  177777  5$:     .WORD   -1,-1,-1,-1      ;ERROR FRACTIONAL RES.
5866  026672  177777
5867  026674  177777  177777  177777  6$:     .WORD   -1,-1,-1,-1      ;ERROR INTEGER RES.
5868  026702  177777
5869  026704  000217                  7$:     217                      ;FPS BEFORE EXECUTION.
5870  026706  000200                          200                      ;FPS AFTER EXECUTION.
```

```
5871  026710  104070                  8$:     ERROR   70
5872  026712  000401                          BR      9$
5873  026714  104071                          ERROR   71
5874  026716                          9$:
5875
5876                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 30
5877  026716                          ####9:
5878  026716  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5879  026720  004737  027422                  JSR     PC,@#MODDSUB
5880  026724  040200  000000  000000  1$:     .WORD   40200,0,0,0     ;AC
5881  026732  000000
5882  026734  047577  177777          2$:     .WORD   47577,-1        ;FSRC
5883  026740  176000  000001                  .WORD   176000,1
5884  026744  031600  000000  000000  3$:     .WORD   31600,0,0,0     ;FRACTIONAL RES.
5885  026752  000000
5886  026754  047577  177777          4$:     .WORD   47577,-1        ;INTEGER RES.
5887  026760  176000  000000                  .WORD   176000,0
5888  026764  000000  000000  000000  5$:     .WORD   0,0,0,0         ;ERROR FRACTIONAL RES.
5889  026772  000000
5890  026774  047577  177777  177777  6$:     .WORD   47577,-1,-1,-1  ;ERROR INTEGER RES.
5891  027002  177777
5892  027004  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
5893  027006  000200                          200                     ;FPS AFTER EXECUTION.
5894  027010  104103                  8$:     ERROR   103             ;(NORMALIZE) ST 532 TO 122
5895                                                                  ;INTO NORM.
5896  027012  000401                          BR      9$
5897  027014  104104                          ERROR 104               ;AC V 1 <= X14
5898                                                                  ;OR ST 733 TO 156 INTO 157.
5899  027016                          9$:
5900
5901                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 31
5902  027016                          ####10:
5903  027016  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
5904  027020  004737  027422                  JSR     PC,@#MODDSUB
5905  027024  047777  177777          1$:     .WORD   47777,-1        ;AC
5906  027030  177000  000000                  .WORD   177000,0
5907  027034  040200  000000  000000  2$:     .WORD   40200,0,0,0     ;FSRC
5908  027042  000000
5909  027044  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
5910  027052  000000
5911  027054  047777  177777          4$:     .WORD   47777,-1        ;INTEGER RES.
5912  027060  177000  000000                  .WORD   177000,0
5913  027064  000000  000000  177000  5$:     .WORD   0,0,177000,0    ;ERROR FRACTIONAL RES.
5914  027072  000000
5915  027074  177777  177777  177777  6$:     .WORD   -1,-1,-1,-1     ;ERROR INTEGER RES.
5916  027102  177777
5917  027104  000213                  7$:     213                     ;FPS BEFORE EXECUTION.
5918  027106  000204                          204                     ;FPS AFTER EXECUTION.
5919  027110  104105                  8$:     ERROR   105             ;(BUT FD) STORE X10
5920  027112  000401                          BR      9$
5921  027114  104071                          ERROR   71
5922  027116                          9$:
5923
5924                                  ;MODD TEST WITH EXPONENT OF THE RESULT = 0
5925  027116                          ####11:
5926  027116  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
```

```
5927  027120  004737  027422                      JSR     PC,@#MODDSUB
5928  027124  040200  000000  000000  1$:         .WORD   40200,0,0,0       ;AC
5929  027132  000000
5930  027134  040177  072727          2$:         .WORD   40177,72727       ;FSRC
5931  027140  127272  072727                      .WORD   127272,72727
5932  027144  040177  072727          3$:         .WORD   40177,72727       ;FRACTIONAL RES.
5933  027150  127272  072727                      .WORD   127272,72727
5934  027154  000000  000000  000000  4$:         .WORD   0,0,0,0           ;INTEGER RES.
5935  027162  000000
5936  027164  177777  177777  177777  5$:         .WORD   -1,-1,-1,-1       ;ERROR FRACTIONAL RES.
5937  027172  177777
5938  027174  000000  000000  177777  6$:         .WORD   0,0,-1,-1         ;ERROR INTEGER RES.
5939  027202  177777
5940  027204  000200                  7$:         200                       ;FPS BEFORE EXECUTION.
5941  027206  000200                              200                       ;FPS AFTER EXECUTION.
5942  027210  104070                  8$:         ERROR   70
5943  027212  000401                              BR      9$
5944  027214  104106                              ERROR   106               ;ST 246 TO 126 INTO 127 (BUT FD)
5945  027216                          9$:
5946
5947                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -115
5948  027216                          HHH12:
5949  027216  104413                              LPERR                     ;SET UP THE LOOP ON ERROR ADDRESS.
5950  027220  004737  027422                      JSR     PC,@#MODDSUB
5951  027224  003377  177777          1$:         .WORD   3377,-1           ;AC
5952  027230  177777  052525                      .WORD   -1,52525
5953  027234  040200  000000  000000  2$:         .WORD   40200,0,0,0       ;FSRC
5954  027242  000000
5955  027244  003377  177777          3$:         .WORD   3377,-1           ;FRACTIONAL RES.
5956  027250  177777  052525                      .WORD   -1,52525
5957  027254  000000  000000  000000  4$:         .WORD   0,0,0,0           ;INTEGER RES.
5958  027262  000000
5959  027264  177777  177777  177777  5$:         .WORD   -1,-1,-1,-1       ;ERROR FRACTIONAL RES.
5960  027272  177777
5961  027274  000000  000000  177777  6$:         .WORD   0,0,-1,-1         ;ERROR INTEGER RES.
5962  027302  177777
5963  027304  000200                  7$:         200                       ;FPS BEFORE EXECUTION.
5964  027306  000200                              200                       ;FPS AFTER EXECUTION.
5965  027310  104070                  8$:         ERROR   70
5966  027312  000401                              BR      9$
5967  027314  104107                              ERROR   107               ;ST 446 TO 126 INTO 127 (BUT FD)
5968  027316                          9$:
5969
5970                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.
5971  027316                          HHH13:
5972  027316  104413                              LPERR                     ;SET UP THE LOOP ON ERROR ADDRESS.
5973  027320  004737  027422                      JSR     PC,@#MODDSUB
5974  027324  040300  000000  000000  1$:         .WORD   40300,0,0,0       ;AC
5975  027332  000000
5976  027334  020200  000000  000000  2$:         .WORD   20200,0,0,1       ;FSRC
5977  027342  000001
5978  027344  020300  000000  000000  3$:         .WORD   20300,0,0,2       ;FRACTIONAL RES.
5979  027352  000002
5980  027354  000000  000000  000000  4$:         .WORD   0,0,0,0           ;INTEGER RES.
5981  027362  000000
5982  027364  000000  000000  177777  5$:         .WORD   0,0,-1,-1         ;ERROR FRACTIONAL RES.
```

```
5983  027372  177777
5984  027374  177777  177777  177777  6$:     .WORD   -1,-1,-1,-1      ;ERROR INTEGER RES.
5985  027402  177777
5986  027404  000200                  7$:     200                       ;FPS BEFORE EXECUTION.
5987  027406  000200                          200                       ;FPS AFTER EXECUTION.
5988  027410  104110                  8$:     ERROR   110               ;ST 127 INTO RND/TR
5989  027412  000401                          BR      9$
5990  027414  104071                          ERROR   71
5991  027416  000137  030020          9$:     JMP     @#HHHDONE         ;GO TO THE NEXT TEST.
5992
5993                                  ;THIS SUBROUTINE, MODDSUB, IS CALLED TO SETUP THE
5994                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
5995                                  ;IT IS CALLED THUS:
5996                                  ;
5997                                  ;               ACARG:   .WORD   X,X,X,X          ;AC OPERAND
5998                                  ;               FSRCARG: .WORD   X,X,X,X          ;FSRC OPERAND
5999                                  ;               FRES:    .WORD   X,X,X,X          ;FRACTIONAL RESULT
6000                                  ;               INTRES:  .WORD   X,X,X,X          ;INTEGER RESULT
6001                                  ;               ERFRES:  .WORD   X,X,X,X          ;ERROR FRACTION RESULT
6002                                  ;               ERINTRES:.WORD   X,X,X,X          ;ERROR INTEGER RESULT
6003                                  ;               FPSB:    .WORD   X                ;FPS BEFORE EXECUTION
6004                                  ;               FPSA:    .WORD   X                ;FPS AFTER EXECUTION
6005                                  ;               ERR1:    ERROR   X                ;FRACTION ERROR
6006                                  ;                        BR      CONT
6007                                  ;               ERR2:    ERROR   X                ;INTEGER ERROR
6008                                  ;               CONT:                             ;RETURN ADDRESS
6009                                  ;
6010                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
6011                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
6012                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
6013                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
6014                                  ;THEN THE FPS IS COMPARED WITH FPSA, AFTER EXECUTION IF NO ERRORS OCCURRED
6015                                  ;THEN MODDSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
6016                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
6017                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
6018                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
6019                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODDSUB PASSES CONTROL TO THE
6020                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
6021                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
6022                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
6023                                  ;IF A MATCH IS MADE HOWEVER, MODDSUB WILL RETURN CONTROL TO THE ERROR
6024                                  ;CALL AT ERR2.
6025
6026  027422  012601                  MODDSUB:        MOV     (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS
6027  027424  012700  000200                  MOV     #200,R0           ;SET FD MODE.
6028  027430  170100                          LDFPS   R0
6029  027432  010100                          MOV     R1,R0             ;SET UP AC0
6030  027434  172410                          LDD     (R0),AC0
6031  027436  012700  025702                  MOV     #MODP1,R0         ;PUT A BACKROUND PATTERN INTO AC1.
6032  027442  172510                          LDD     (R0),AC1
6033  027444  016100  000060                  MOV     60(R1),R0         ;SET UP THE FPS.
6034  027450  170100                          LDFPS   R0
6035  027452  012737  027466  001236          MOV     #1$,@#$TMP2
6036  027460  010100                          MOV     R1,R0             ;COMPUTE THE ADDRESS OF THE FSRC.
6037  027462  062700  000010                  ADD     #10,R0
6038
```

```
6039  027466  171410                  1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
6040
6041  027470  170204                          STFPS   R4              ;GET THE FPS.
6042  027472  012700  000200                  MOV     #200,R0         ;SET FD MODE.
6043  027476  170100                          LDFPS   R0
6044  027500  012700  030000                  MOV     #MODDT0,R0      ;GET THE FRACTIONAL RESULT.
6045  027504  174010                          STD     AC0,(R0)
6046  027506  012700  030010                  MOV     #MODDT1,R0      ;GET THE INTEGER RESULT.
6047  027512  174110                          STD     AC1,(R0)
6048
6049  027514  010102                          MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
6050  027516  010237  001240                  MOV     R2,@#$TMP3
6051  027522  062702  000010                  ADD     #10,R2
6052  027526  010237  001242                  MOV     R2,@#$TMP4
6053  027532  062702  000010                  ADD     #10,R2
6054  027536  010237  001244                  MOV     R2,@#$TMP5
6055  027542  062702  000010                  ADD     #10,R2
6056  027546  010237  001246                  MOV     R2,@#$TMP6
6057  027552  012737  030000  001250          MOV     #MODDT0,@#$TMP7
6058  027560  012737  030010  001252          MOV     #MODDT1,@#$TMP10
6059  027566  016137  000062  001256          MOV     62(R1),@#$TMP12
6060  027574  010437  001254                  MOV     R4,@#$TMP11
6061
6062  027600  012702  030000                  MOV     #MODDT0,R2      ;CHECK THE FRACTIONAL RESULT.
6063  027604  010103                          MOV     R1,R3
6064  027606  062703  000020                  ADD     #20,R3
6065  027612  012705  000004                  MOV     #4,R5
6066  027616  022223                  2$:     CMP     (R2)+,(R3)+
6067  027620  001020                          BNE     10$             ;BRANCH IF INCORRECT.
6068  027622  077503                          SOB     R5,2$
6069
6070  027624  012702  030010                  MOV     #MODDT1,R2      ;CHECK THE INTEGER RESULT.
6071  027630  010103                          MOV     R1,R3
6072  027632  062703  000030                  ADD     #30,R3
6073  027636  012705  000004                  MOV     #4,R5
6074  027642  022223                  3$:     CMP     (R2)+,(R3)+
6075  027644  001026                          BNE     15$             ;BRANCH IF INCORRECT.
6076  027646  077503                          SOB     R5,3$
6077
6078
6079  027650  026104  000062                  CMP     62(R1),R4       ;CHECK THE FPS.
6080  027654  001042                          BNE     20$             ;BRANCH IF INCORRECT.
6081
6082  027656  000161  000072          9$:     JMP     72(R1)          ;RETURN.
6083
6084                                  ;FRACTIONAL ERROR.
6085  027662  012702  030000          10$:    MOV     #MODDT0,R2      ;WAS THE FRACTIONAL ERROR ANTICIPATED?
6086  027666  010103                          MOV     R1,R3
6087  027670  062703  000040                  ADD     #40,R3
6088  027674  012705  000004                  MOV     #4,R5
6089  027700  022223                  50$:    CMP     (R2)+,(R3)+
6090  027702  001005                          BNE     11$             ;BRANCH IF NOT ANTICIPATED.
6091  027704  077503                          SOB     R5,50$
6092  027706  010102                          MOV     R1,R2           ;THE ERROR WAS ANTICIPATED SO
6093  027710  062702  000064                  ADD     #64,R2          ;RETURN TO THE ERROR REPORT AT THE
6094                                                                  ;CALLING ROUTINE.
```

```
6095  027714  000112                          JMP     (R2)
6096
6097  027716                          11$:                            ;THE ERROR WAS NOT ANTICIPATED SO
6098  027716  104070                  12$:    ERROR   70              ;REPORT THE INCORRECT FRACTION HERE.
6099  027720  000756                          BR      9$
6100
6101                                  ;INTEGER ERROR.
6102  027722  012702  030010          15$:    MOV     #MODDT1,R2      ;WAS THE INTEGER ERROR ANTICIPATED
6103  027726  010103                          MOV     R1,R3
6104  027730  062703  000050                  ADD     #50,R3
6105  027734  012705  000004                  MOV     #4,R5
6106  027740  022223                  60$:    CMP     (R2)+,(R3)+
6107  027742  001005                          BNE     17$             ;BRANCH IF NOT ANTICIPATED.
6108  027744  077503                          SOB     R5,60$
6109  027746  010102                          MOV     R1,R2           ;THE ERROR WAS ANTICIAPTED SO RETURN
6110  027750  062702  000070                  ADD     #70,R2          ;TO THE ERROR REPORT IN THE CALLING
6111                                                                  ;ROUTINE.
6112  027754  000112                          JMP     (R2)
6113
6114  027756                          16$:                            ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6115  027756  104071                  17$:    ERROR   71              ;THE INTEGER FAILURE HERE.
6116  027760  000736                          BR      9$
6117
6118                                  ;FPS INCORRECT.
6119  027762  010437  001254          20$:    MOV     R4,@#$TMP11     ;REPORT INCORRECT FPS.
6120  027766  016137  000062  001256          MOV     62(R1),@#$TMP12
6121  027774  104072                  21$:    ERROR   72
6122  027776  000727                          BR      9$
6123
6124  030000  000000  000000  000000  MODDT0: .WORD   0,0,0,0
6125  030006  000000
6126
6127  030010  000000  000000  000000  MODDT1: .WORD   0,0,0,0
6128  030016  000000
6129
6130  030020                          HHHDONE:
6131  030020  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
6132                                                                  ;SEE IF THE USER HAS EXPRESSED
6133                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
6134                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6135                                                                  ;THE USER TYPED CONTROL G?).
6136
6137
6138
6139
6140                                  ;;*****************************************************************
6141                                  ;*TEST 20       UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST
6142                                  ;*
6143                                  ;*THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES
6144                                  ;*USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
6145                                  ;*AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.
6146                                  ;*
6147                                  ;*****************************************************************
6148  030022  000004                  TST20:  SCOPE
6149
6150                                  ;UNDERFLOW TEST, WITH EXPONENT OF THE RESULT = -129, FIU = 1, FIC = 1
```

```
6151  030024                          MMM1:
6152  030024  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6153  030026  004767  000320                  JSR     PC,MODFOV
6154  030032  020123  045676          1$:     .WORD   20123,45676     ;AC
6155  030036  020200  000000          2$:     .WORD   20200,0         ;FSRC
6156  030042  000123  045676          3$:     .WORD   123,45676       ;FRACTIONAL RES.
6157  030046  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
6158  030052  177777  177777          5$:     .WORD   -1,-1           ;ERROR FRACTIONAL RES.
6159  030056  177777  177777          6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
6160  030062  042000                  7$:     42000                   ;FPS BEFORE EXECUTION.
6161  030064  142004                          142004                  ;FPS AFTER EXECUTION.
6162  030066  000012                          12                      ;FEC
6163  030070  104170                  8$:     ERROR   170             ;FEC INCORRECT, UNDERFLOW.
6164  030072  000401                          BR      9$
6165  030074  104171                          ERROR   171             ;AC V 1 (2,3) <= ZERO, ST 128.
6166  030076                          9$:
6167
6168                                  ;UNDERFLOW EXP OF RESULT = -193, FIU = 0, FID = 1
6169  030076                          MMM2:
6170  030076  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6171  030100  004737  030352                  JSR     PC,@#MODFOV
6172  030104  010200  000000          1$:     .WORD   10200,0         ;AC
6173  030110  010000  000000          2$:     .WORD   10000,0         ;FSRC
6174  030114  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
6175  030120  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
6176  030124  177777  177777          5$:     .WORD   -1,-1           ;ERROR FRACTIONAL RES.
6177  030130  177777  177777          6$:     .WORD   -1,-1           ;ERROR INTEGER RES.
6178  030134  005013                  7$:     5013                    ;FPS BEFORE EXECUTION.
6179  030136  005004                          5004                    ;FPS AFTER EXECUTION.
6180  030140  000012                          12                      ;FEC
6181  030142  000240                  8$:     NOP
6182  030144  000401                          BR      9$
6183  030146  104171                          ERROR   171
6184  030150                          9$:
6185
6186                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
6187  030150                          MMM3:
6188  030150  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6189  030152  004737  030352                  JSR     PC,@#MODFOV
6190  030156  060052  125252          1$:     .WORD   60052,125252    ;AC
6191  030162  060200  000000          2$:     .WORD   60200,0         ;FSRC
6192  030166  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
6193  030172  000052  125252          4$:     .WORD   52,125252       ;INTEGER RES.
6194  030176  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
6195  030202  000000  000000          6$:     .WORD   0,0             ;ERROR INTEGER RES.
6196  030206  041000                  7$:     41000                   ;FPS BEFORE EXECUTION.
6197  030210  141006                          141006                  ;FPS AFTER EXECUTION.
6198  030212  000010                          10                      ;FEC
6199  030214  104172                  8$:     ERROR   172             ;BAD FEC ON OVERFLOW.
6200  030216  000401                          BR      9$
6201  030220  104173                          ERROR   173             ;ST 520 TO STORE ZERO TWICE
6202                                                                  ;INTO 162
6203  030222                          9$:
6204
6205                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
6206  030222                          MMM4:
```

```
6207  030222  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6208  030224  004737  030352                  JSR     PC,@#MODFOV
6209  030230  060345  067654          1$:     .WORD   60345,67654     ;AC
6210  030234  060200  000000          2$:     .WORD   60200,0         ;FSRC
6211  030240  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
6212  030244  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
6213  030250  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
6214  030254  000345  067654          6$:     .WORD   345,67654       ;ERROR INTEGER RES.
6215  030260  006011                  7$:     6011                    ;FPS BEFORE EXECUTION.
6216  030262  006006                          6006                    ;FPS AFTER EXECUTION.
6217  030264  000010                          10                      ;FEC
6218  030266  000240                  8$:     NOP
6219  030270  000401                          BR      9$
6220  030272  104174                          ERROR   174             ;ST 520 TO 162 INTO STORE ZERO TWICE.
6221  030274                          9$:
6222
6223                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
6224                                  ;AND FIV = 1, FID = 1
6225  030274                          MMM5:
6226  030274  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6227  030276  004737  030352                  JSR     PC,@#MODFOV
6228  030302  160252  125252          1$:     .WORD   160252,125252   ;AC
6229  030306  060000  000000          2$:     .WORD   60000,0         ;FSRC
6230  030312  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
6231  030316  100052  125252          4$:     .WORD   100052,125252   ;INTEGER RES.
6232  030322  000000  000000          5$:     .WORD   0,0             ;ERROR FRACTIONAL RES.
6233  030326  000052  125252          6$:     .WORD   52,125252       ;ERROR INTEGER RES.
6234  030332  041000                  7$:     41000                   ;FPS BEFORE EXECUTION.
6235  030334  141006                          141006                  ;FPS AFTER EXECUTION.
6236  030336  000010                          10                      ;FEC
6237  030340  104172                  8$:     ERROR   172
6238  030342  000401                          BR      9$
6239  030344  104175                          ERROR   175             ;ST 517, BAD SIGN.
6240  030346  000137  030746          9$:     JMP     @#MMMDONE       ;GO TO THE NEXT TEST.
6241
6242                                  ;THIS SUBROUTINE, MODFOV, IS CALLED TO SETUP THE
6243                                  ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
6244                                  ;IT IS CALLED THUS:
6245                                  ;
6246                                  ;               ACARG:   .WORD   X,X            ;AC OPERAND
6247                                  ;               FSRCARG:.WORD    X,X            ;FSRC OPERAND
6248                                  ;               FRES:    .WORD   X,X            ;FRACTIONAL RESULT
6249                                  ;               INTRES:  .WORD   X,X            ;INTEGER RESULT
6250                                  ;               ERFRES:  .WORD   X,X            ;ERROR FRACTION RESULT
6251                                  ;               ERINTRES:.WORD   X,X            ;ERROR INTEGER RESULT
6252                                  ;               FPSB:    .WORD   X              ;FPS BEFORE EXECUTION
6253                                  ;               FPSA:    .WORD   X              ;FPS AFTER EXECUTION
6254                                  ;               FEC:     .WORD   X              ;FEC
6255                                  ;               ERR1:    ERROR   X              ;FEC ERROR
6256                                  ;                        BR      CONT
6257                                  ;               ERR2:    ERROR   X              ;INTEGER ERROR
6258                                  ;               CONT:                           ;RETURN ADDRESS
6259                                  ;
6260                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
6261                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
6262                                  ;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.

030352  012601          MODFOV: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
030354  012700  000200          MOV     #200,R0         ;SET FD MODE.
030360  170100                  LDFPS   R0
030362  010100                  MOV     R1,R0           ;SET UP AC0
030364  172410                  LDD     (R0),AC0
030366  012700  025702          MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
030372  172510                  LDD     (R0),AC1
030374  016100  000030          MOV     30(R1),R0       ;SET UP THE FPS.
030400  170100                  LDFPS   R0
030402  012737  030416  001236  MOV     #1$,@#$TMP2
030410  010100                  MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
030412  062700  000004          ADD     #4,R0

030416  171410          1$:     MODF    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.

030420  170204                  STFPS   R4              ;GET THE FPS.
030422  170305                  STST    R5              ;GET FEC.
030424  012700  000200          MOV     #200,R0         ;SET FD MODE.
030430  170100                  LDFPS   R0
030432  012700  030726          MOV     #MODFD0,R0      ;GET THE FRACTIONAL RESULT.
030436  174010                  STD     AC0,(R0)
030440  012700  030736          MOV     #MODFD1,R0      ;GET THE INTEGER RESULT.
030444  174110                  STD     AC1,(R0)

030446  010102                  MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
030450  010237  001240          MOV     R2,@#$TMP3
030454  062702  000004          ADD     #4,R2
030460  010237  001242          MOV     R2,@#$TMP4
030464  062702  000004          ADD     #4,R2
030470  010237  001244          MOV     R2,@#$TMP5
030474  062702  000004          ADD     #4,R2
030500  010237  001246          MOV     R2,@#$TMP6
030504  012737  030726  001250  MOV     #MODFD0,@#$TMP7
030512  012737  030736  001252  MOV     #MODFD1,@#$TMP10
030520  010437  001254          MOV     R4,@#$TMP11
030524  016137  000032  001256  MOV     32(R1),@#$TMP12
030532  010537  001260          MOV     R5,@#$TMP13
030536  016137  000034  001262  MOV     34(R1),@#$TMP14

030544  012702  030726          MOV     #MODFD0,R2      ;CHECK THE FRACTIONAL RESULT.
030550  026112  000010          CMP     10(R1),(R2)
030554  001025                  BNE     10$             ;BRANCH IF INCORRECT.
030556  026162  000012  000002  CMP     12(R1),2(R2)
```

```
6319  030564  001021                          BNE     10$
6320
6321  030566  012702  030736                  MOV     #MODFD1,R2      ;CHECK THE INTEGER RESULT.
6322  030572  026112  000014                  CMP     14(R1),(R2)
6323  030576  001016                          BNE     15$             ;BRANCH IF INCORRECT.
6324  030600  026162  000016  000002          CMP     16(R1),2(R2)
6325  030606  001012                          BNE     15$
6326
6327  030610  026104  000032                  CMP     32(R1),R4       ;CHECK THE FPS.
6328  030614  001024                          BNE     20$             ;BRANCH IF INCORRECT.
6329
6330  030616  026105  000034                  CMP     34(R1),R5       ;CHECK THE FEC.
6331  030622  001030                          BNE     25$             ;BRANCH IF INCORRECT.
6332
6333  030624  000161  000044          9$:     JMP     44(R1)          ;RETURN.
6334
6335                                  ;FRACTIONAL ERROR.
6336  030630                          10$:                            ;THE ERROR WAS NOT ANTICIPATED SO
6337  030630  104165                  12$:    ERROR   165             ;REPORT THE INCORRECT FRACTION HERE.
6338  030632  000774                          BR      9$
6339
6340                                  ;INTEGER ERROR.
6341  030634  026112  000024          15$:    CMP     24(R1),(R2)     ;WAS THIS ERROR ANTICIPATED?
6342  030640  001010                          BNE     16$             ;BRANCH IF NOT.
6343  030642  026162  000026  000002          CMP     26(R1),2(R2)
6344  030650  001004                          BNE     16$
6345  030652  010102                          MOV     R1,R2           ;THE ERROR WAS ANTICIAPTED SO RETURN
6346  030654  062702  000042                  ADD     #42,R2          ;TO THE ERROR REPORT IN THE CALLING
6347                                                                  ;ROUTINE.
6348  030660  000112                          JMP     (R2)
6349
6350  030662                          16$:                            ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6351  030662  104166                  17$:    ERROR   166             ;THE INTEGER FAILURE HERE.
6352  030664  000757                          BR      9$
6353
6354                                  ;FPS INCORRECT.
6355  030666  010437  001254          20$:    MOV     R4,@#$TMP11     ;REPORT INCORRECT FPS.
6356  030672  016137  000032  001256          MOV     32(R1),@#$TMP12
6357  030700  104167                  21$:    ERROR   167
6358  030702  000750                          BR      9$
6359
6360                                  ;REPORT FEC ERROR.
6361  030704  010537  001260          25$:    MOV     R5,@#$TMP13
6362  030710  016137  000034  001262          MOV     34(R1),@#$TMP14
6363  030716  010102                          MOV     R1,R2
6364  030720  062702  000036                  ADD     #36,R2
6365  030724  000112                          JMP     (R2)
6366
6367  030726  000000  000000  000000  MODFD0: .WORD   0,0,0,0
6368  030734  000000
6369
6370  030736  000000  000000  000000  MODFD1: .WORD   0,0,0,0
6371  030744  000000
6372
6373  030746                          MMMDONE:
6374  030746  104412                          RSETUP                  ;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 21       UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST
                                ;*
                                ;*THIS IS A TEST OF THE MODD INSTRUCTION'S OVER FLOW AND UNDER FLOW
                                ;*CONDITIONS. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE
                                ;*MODD INSTUCTION AND CHECK THE RESULTS.
                                ;*
                                ;;*****************************************************************
030750  000004                  TST21:  SCOPE

                                ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
030752                          NNN1:
030752  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
030754  004737  031366                  JSR     PC,@#MODDOV
030760  020252  125252          1$:     .WORD   20252,125252    ;AC
030764  125252  125252                  .WORD   125252,125252
030770  020100  000000  000000  2$:     .WORD   20100,0,0,0     ;FSRC
030776  000000
031000  000177  177777  177777  3$:     .WORD   177,-1,-1,-1    ;FRACTIONAL RES.
031006  177777
031010  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
031016  000000
031020  020252  125252          5$:     .WORD   20252,125252    ;ERROR FRACTIONAL RES.
031024  125252  125252                  .WORD   125252,125252
031030  000000  000000  177777  6$:     .WORD   0,0,-1,-1       ;ERROR INTEGER RES.
031036  177777
031040  042200                  7$:     42200                   ;FPS BEFORE EXECUTION.
031042  142204                          142204                  ;FPS AFTER EXECUTION.
031044  000012                          12                      ;FEC
031046  104201                  8$:     ERROR   201             ;FEC INCORRECT ON UNDERFLOW.
031050  000401                          BR      9$
031052  104202                          ERROR   202             ;ST 155 (BUT FD)
031054                          9$:

                                ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -193, FIU = 0, FID = 1
031054                          NNN2:
031054  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
031056  004737  031366                  JSR     PC,@#MODDOV
031062  010000  000000          1$:     .WORD   10000,0         ;AC
031066  123456  000000                  .WORD   123456,0
031072  010200  000000  000000  2$:     .WORD   10200,0,0,0     ;FSRC
031100  000000
031102  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
031110  000000
031112  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
031120  000000
031122  000000  000000  000000  5$:     .WORD   0,0,0,0         ;ERROR FRACTIONAL RES.
031130  000000
```

```
6431  031132  000000  000000          6$:     .WORD   0,0             ;ERROR INTEGER RES.
6432  031136  123456  000000                  .WORD   123456,0
6433  031142  005213                  7$:     5213                    ;FPS BEFORE EXECUTION.
6434  031144  005204                          5204                    ;FPS AFTER EXECUTION.
6435  031146  000012                          12
6436  031150  000240                  8$:     NOP
6437  031152  000401                          BR      9$
6438  031154  104203                          ERROR   203             ;ST 047 (BUT FD).
6439  031156                          9$:
6440
6441                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
6442  031156                          NNN3:
6443  031156  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6444  031160  004737  031366                  JSR     PC,@#MODDOV
6445  031164  060252  125252          1$:     .WORD   60252,125252    ;AC
6446  031170  125252  125252                  .WORD   125252,125252
6447  031174  060100  000000  000000  2$:     .WORD   60100,0,0,0     ;FSRC
6448  031202  000000
6449  031204  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
6450  031212  000000
6451  031214  000177  177777  177777  4$:     .WORD   177,-1,-1,-1    ;INTEGER RES.
6452  031222  177777
6453  031224  000000  000000  000000  5$:     .WORD   0,0,0,0         ;ERROR FRACTIONAL RES.
6454  031232  000000
6455  031234  000177  177777          6$:     .WORD   177,-1          ;ERROR INTEGER RES.
6456  031240  125252  125252                  .WORD   125252,125252
6457  031244  041200                  7$:     41200                   ;FPS BEFORE EXECUTION.
6458  031246  141206                          141206                  ;FPS AFTER EXECUTION.
6459  031250  000010                          10                      ;FEC
6460  031252  104204                  8$:     ERROR   204             ;FEC BAD ON OVERFLOW.
6461  031254  000401                          BR      9$
6462  031256  104205                          ERROR   205             ;ST 520 TO 162 INTO 163 (BUT FD).
6463  031260                          9$:
6464
6465                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
6466  031260                          NNN4:
6467  031260  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6468  031262  004737  031366                  JSR     PC,@#MODDOV
6469  031266  060200  000000          1$:     .WORD   60200,0         ;AC
6470  031272  125252  000000                  .WORD   125252,0
6471  031276  060200  000000  000000  2$:     .WORD   60200,0,0,0     ;FSRC
6472  031304  000000
6473  031306  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
6474  031314  000000
6475  031316  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
6476  031324  000000
6477  031326  000000  000000  000000  5$:     .WORD   0,0,0,0         ;ERROR FRACTIONAL RES.
6478  031334  000000
6479  031336  000400  000000          6$:     .WORD   400,0           ;ERROR INTEGER RES.
6480  031342  125252  000000                  .WORD   125252,0
6481  031346  006211                  7$:     6211                    ;FPS BEFORE EXECUTION.
6482  031350  006206                          6206                    ;FPS AFTER EXECUTION.
6483  031352  000010                          10                      ;FEC
6484  031354  000240                  8$:     NOP
6485  031356  000401                          BR      9$
6486  031360  104206                          ERROR   206             ;ST 520 TO 162 INTO STORE ZERO TWICE.
```

```
 6487  001362  000137  031774          9$:     JMP     @#NNNDONE       ;GO TO NEXT TEST.
 6488
 6489                                  ;THIS SUBROUTINE, MODDOV, IS CALLED TO SETUP THE
 6490                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
 6491                                  ;IT IS CALLED THUS:
 6492                                  ;
 6493                                  ;               ACARG:   .WORD   X,X,X,X         ;AC OPERAND
 6494                                  ;               FSRCARG:.WORD    X,X,X,X         ;FSRC OPERAND
 6495                                  ;               FRES:    .WORD   X,X,X,X         ;FRACTIONAL RESULT
 6496                                  ;               INTRES:  .WORD   X,X,X,X         ;INTEGER RESULT
 6497                                  ;               ERFRES:  .WORD   X,X,X,X         ;ERROR FRACTION RESULT
 6498                                  ;               ERINTRES:.WORD   X,X,X,X         ;ERROR INTEGER RESULT
 6499                                  ;               FPSB:    .WORD   X               ;FPS BEFORE EXECUTION
 6500                                  ;               FPSA:    .WORD   X               ;FPS AFTER EXECUTION
 6501                                  ;               ERR1:    ERROR   X               ;FRACTION ERROR
 6502                                  ;                        BR      CONT
 6503                                  ;               ERR2:    ERROR   X               ;INTEGER ERROR
 6504                                  ;               CONT:                            ;RETURN ADDRESS
 6505                                  ;
 6506                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
 6507                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
 6508                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
 6509                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
 6510                                  ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
 6511                                  ;THEN MODDOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
 6512                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
 6513                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
 6514                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
 6515                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODDOV PASSES CONTROL TO THE
 6516                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
 6517                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
 6518                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
 6519                                  ;IF A MATCH IS MADE HOWEVER, MODDOV WILL RETURN CONTROL TO THE ERROR
 6520                                  ;CALL AT ERR2.
 6521
 6522  031366  012601                  MODDOV: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
 6523  031370  012700  000200                  MOV     #200,R0         ;SET FD MODE.
 6524  031374  170100                          LDFPS   R0
 6525  031376  010100                          MOV     R1,R0           ;SET UP AC0
 6526  031400  172410                          LDD     (R0),AC0
 6527  031402  012700  025702                  MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
 6528  031406  172510                          LDD     (R0),AC1
 6529  031410  016100  000060                  MOV     60(R1),R0       ;SET UP THE FPS.
 6530  031414  170100                          LDFPS   R0
 6531  031416  012737  031432  001236          MOV     #1$,@#$TMP2
 6532  031424  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
 6533  031426  062700  000010                  ADD     #10,R0
 6534
 6535  031432  171410                  1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
 6536
 6537  031434  170305                          STST    R5              ;GET THE FPS.
 6538  031436  170204                          STFPS   R4              ;GET THE FPS.
 6539  031440  012700  000200                  MOV     #200,R0         ;SET FD MODE.
 6540  031444  170100                          LDFPS   R0
 6541  031446  012700  031754                  MOV     #MODDDD,R0      ;GET THE FRACTIONAL RESULT.
 6542  031452  174010                          STD     AC0,(R0)
```

```
6543  031454  012700  031764          MOV     #MODDD1,R0      ;GET THE INTEGER RESULT.
6544  031460  174110                  STD     AC1,(R0)
6545
6546  031462  010102                  MOV     R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
6547  031464  010237  001240          MOV     R2,@#$TMP3
6548  031470  062702  000010          ADD     #10,R2
6549  031474  010237  001242          MOV     R2,@#$TMP4
6550  031500  062702  000010          ADD     #10,R2
6551  031504  010237  001244          MOV     R2,@#$TMP5
6552  031510  062702  000010          ADD     #10,R2
6553  031514  010237  001246          MOV     R2,@#$TMP6
6554  031520  012737  031754  001250  MOV     #MODDD0,@#$TMP7
6555  031526  012737  031764  001252  MOV     #MODDD1,@#$TMP10
6556  031534  010437  001254          MOV     R4,@#$TMP11
6557  031540  016137  000062  001256  MOV     62(R1),@#$TMP12
6558  031546  010537  001260          MOV     R5,@#$TMP13
6559  031552  016137  000064  001262  MOV     64(R1),@#$TMP14
6560
6561  031560  012702  031754          MOV     #MODDD0,R2      ;CHECK THE FRACTIONAL RESULT.
6562  031564  010103                  MOV     R1,R3
6563  031566  062703  000020          ADD     #20,R3
6564  031572  012700  000004          MOV     #4,R0
6565  031576  022223          2$:     CMP     (R2)+,(R3)+
6566  031600  001023                  BNE     10$             ;BRANCH IF INCORRECT.
6567  031602  077003                  SOB     R0,2$
6568
6569  031604  012702  031764          MOV     #MODDD1,R2      ;CHECK THE INTEGER RESULT.
6570  031610  010103                  MOV     R1,R3
6571  031612  062703  000030          ADD     #30,R3
6572  031616  012700  000004          MOV     #4,R0
6573  031622  022223          3$:     CMP     (R2)+,(R3)+
6574  031624  001013                  BNE     15$             ;BRANCH IF INCORRECT.
6575  031626  077003                  SOB     R0,3$
6576
6577
6578  031630  026104  000062          CMP     62(R1),R4       ;CHECK THE FPS.
6579  031634  001027                  BNE     20$             ;BRANCH IF INCORRECT.
6580
6581  031636  026105  000064          CMP     64(R1),R5       ;CHECK THE FEC.
6582  031642  001033                  BNE     25$
6583
6584  031644  000161  000074  9$:     JMP     74(R1)          ;RETURN.
6585
6586                          ;FRACTIONAL ERROR.
6587  031650                  10$:                            ;THE ERROR WAS NOT ANTICIPATED SO
6588  031650  104176          12$:    ERROR   176             ;REPORT THE INCORRECT FRACTION HERE.
6589  031652  000774                  BR      9$
6590
6591                          ;INTEGER ERROR.
6592  031654  012702  031764  15$:    MOV     #MODDD1,R2      ;WAS THE INTEGER ERROR ANTICIPATED?
6593  031660  010103                  MOV     R1,R3
6594  031662  062703  000050          ADD     #50,R3
6595  031666  012705  000004          MOV     #4,R5
6596  031672  022223          60$:    CMP     (R2)+,(R3)+
6597  031674  001005                  BNE     17$             ;BRANCH IF NOT ANTICIPATED.
6598  031676  077503                  SOB     R5,60$
```

```
6599  031700  010102                          MOV     R1,R2           ;THE ERROR WAS ANTICIPATED SO RETURN
6600  031702  062702  000072                  ADD     #72.R2          ;TO THE ERROR REPORT IN THE CALLING
6601                                                                  ;ROUTINE.
6602  031706  000112                          JMP     (R2)
6603
6604  031710                          16$:                            ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6605  031710  104177                  17$:    ERROR   177             ;THE INTEGER FAILURE HERE.
6606  031712  000754                          BR      9$
6607
6608                                  ;FPS INCORRECT.
6609  031714  010437  001254          20$:    MOV     R4,@#STMP11     ;REPORT INCORRECT FPS.
6610  031720  016137  000062  001256          MOV     62(R1),@#STMP12
6611  031726  104200                  21$:    ERROR   200
6612  031730  000745                          BR      9$
6613
6614                                  ;REPORT FEC ERROR.
6615  031732  010537  001260          25$:    MOV     R5,@#STMP13
6616  031736  016137  000064  001262          MOV     64(R1),@#STMP14
6617  031744  010102                          MOV     R1,R2
6618  031746  062702  000066                  ADD     #66,R2
6619  031752  000112                          JMP     (R2)
6620
6621  031754  000000  000000  000000  MODD0:  .WORD   0,0,0,0
6622  031762  000000
6623
6624  031764  000000  000000  000000  MODD1:  .WORD   0,0,0,0
6625  031772  000000
6626
6627  031774                          MMNDONE:
6628  031774  104412                          RSETUP                  ;GO INITIALIZE THE FPS AND STACK; AND
6629                                                                  ;SEE IF THE USER HAS EXPRESSED
6630                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
6631                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6632                                                                  ;THE USER TYPED CONTROL G?).
6633
6634
6635                                  ;*****************************************************************
6636                                  ;*TEST 22       INTERRUPT CORRECT FLOWS TEST
6637                                  ;*
6638                                  ;*THIS IS A TEST OF THE 'CORRECT' FLOWS. THIS PART OF THE MICRO CODE
6639                                  ;*HAS AS ITS PURPOSE INSURING THAT INTERRUPT REQUESTS MADE DURING
6640                                  ;*CERTAIN LENGTHY FPP INSTRUCTIONS GET HONORED. THIS IS DONE
6641                                  ;*IN A WAY SUCH THAT IF AN INTERRUPT REQUEST OCCURS DURING ONE
6642                                  ;*OF THESE INSTRUCTIONS THE STATE OF THAT INSTRUCTION'S
6643                                  ;*EXECUTION WILL BE THE SAME AS IF THAT INSTRUCTION HAD NEVER
6644                                  ;*BEEN FETCHED AND ITS EXECUTION NEVER STARTED. THUS THE MICRO CODE
6645                                  ;*WILL RESTORE ALL REGISTERS, BACK UP THE PC AND LEAVE THE
6646                                  ;*FPS AND AC0 THROUGH AC5 UNMODIFIED.
6647                                  ;*THE INSTRUCTIONS FOR WHICH THIS IS NECESSARY ARE:
6648                                  ;*      ADD (OR SUB)
6649                                  ;*      DIV
6650                                  ;*      MUL
6651                                  ;*      MOD
6652                                  ;*(BOTH DOUBLE AND FLOATING)
6653                                  ;*ALL ADDRESSING MODES WILL BE TRIED WITH THE ADDD INSTRUCTION. THEN
6654                                  ;*EACH OF THE OTHER INSTRUCTIONS WILL BE TRIED USING MODE 1.
```

```
                                        ;*NOTE THAT THIS TEST NEEDS A SPECIAL INTERRUPT MODULE,
                                        ;*WHICH WILL PROBABLY ONLY BE PRESENT IN DEC'S MANUFACTURING ENVIPONMENT.
                                        ;*TO RUN. THIS SPECIAL EQUIPMENT IS DESIGNED TO RAISE AN
                                        ;*INTERRUPT REQUEST IN THE PROCESSOR IF A BIT IS SET IN ITS STATUS
                                        ;*REGISTER AND ONLY WHEN AN FPP INSTRUCTION IS ENCOUNTERED.
                                        ;*THEREFORE THIS TEST WILL BE RUN CONDITIONALLY (DEPENDENT UPON WHETHER
                                        ;*OR NOT THE STATUS REGISTER OF THE TEST EQUIPMENT TIMES OUT WHEN REFERENCED.
                                        ;*THIS TEST CAN ALSO BE DESELECTED BY TURNING SWITCH 7 OF THE
                                        ;*SWITCH REGISTER (PHYSICAL OR VIRTUAL) ON.
                                        ;*THE TEST ASSUMES THAT THE TEST EQUIPMENT'S STATUS REGISTER IS AT
                                        ;*LOCATION 777774 (NOTE THAT ALL REFERENCES TO THIS LOCATION ARE
                                        ;*MADE INDIRECT THROUGH THIS PROGRAMS LOCATION CORINT, SO THAT
                                        ;*IF THE USER HAS MODIFIED THE TEST EQUIPMENT'S STATUS REGISTER TO
                                        ;*RESPOND TO A DIFFERENT ADDRESS LOCATION CORINT MUST BE
                                        ;*MADE TO CONTAIN THAT STATUS REGISTER'S NEW ADDRESS).
                                        ;*THIS PROGRAM ASSUMES THAT THE TRAP VECTOR FOR THE TEST EQUIPMENT IS 11C.
                                        ;*AGAIN NOTE THAT ALL REFERENCES TO THIS TRAP VECTOR ARE INDIRECT, THROUGH
                                        ;*THIS PROGRAM'S LOCATION CORTRP (IF THE TEST EQUIPMENT IS
                                        ;*MADE TO TRAP TO A DIFFERENT VECTOR LOCATION CORTRP MUST CONTAIN THE
                                        ;*ADDRESS OF THIS VECTOR).
                                        ;*
                                        ;;*****************************************************************
031776  000004                TST22:  SCOPE

032000  032777  000200  147132        BIT     #200,@SWR       ;SEE IF THE USER HAS DESELECTED THIS
                                                                ;TEST USING THE SWITCH REGISTER.
032006  001402                        BEQ     COR1            ;IF NOT SEE IF TEST EQUIPMENT IS PRESENT.
032010  000137  033146                JMP     @#CORDONE       ;ELSE DO NOT RUN TEST.

032014  012737  032040  000004  COR1:  MOV     #COR2,@#ERRVECT ;SEE IF THE TEST EQUIPMENT'S STATUS
032022  012777  000000  001112        MOV     #0,@CORINT      ;REGISTER TIMES OUT.
032030  012737  036646  000004        MOV     #CPSPUR,@#ERRVECT
032036  000406                        BR      COR3            ;DIDN'T TIME OUT SO START TEST.

032040  022626            COR2:       CMP     (SP)+,(SP)+     ;IF THE REFERENCE TIMES OUT DO
032042  012737  036646  000004        MOV     #CPSPUR,@#ERRVECT
032050  000137  033146                JMP     @#CORDONE       ;NOT RUN TEST.

                                ;TEST ADDO MODE 0
032054                          COR3:
032054  005227  177777                INC     #-1
032060  001002                        BNE     COR33
032062  104401                        TYPE
032064  037621                        .WORD   CORMES
032066                          COR33:
032066  104413                        LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
032070  004737  032646                JSR     PC,@#CORSUB
032074  040200  000100  000200  1$:   .WORD   40200,100,200,300       ;AC0
032102  000300
032104  123456                  2$:   .WORD   123456          ;R0
032106  000200                  3$:   200                     ;FPS
032110  172000                  4$:   ADDO    AC0,AC0         ;TEST INSTRUCTION.
032112  000240                        NOP
032114  005037  033130                CLR     @#CORFLG        ;RESET INTERRUPT FLAG
032120  104252                        ERROR   252             ;NO INTERRUPT! TEST EQUIPMENT FAILED.
032122  000401                        BR      11$
```

```
6711  032124  104253                    5$:     ERROR   253             ;INCORRECT STATE AT INTERRUPT.
6712  032126                            11$:
6713                                    ;TEST ADD MODE 1
6714  032126                            COR4:
6715  032126  104413                            LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6716  032130  004737  032646                    JSR     PC,@#CORSUB
6717  032134  040201  000555  077007  1$:      .WORD   40201,555,77007,111111  ;AC0
6718  032142  111111
6719  032144  032134                    2$:     .WORD   1$              ;R0
6720  032146  000217                    3$:     217                     ;FPS
6721  032150  172010                    4$:     ADDD    (R0),AC0        ;TEST INSTRUCTION
6722  032152  000240                            NOP
6723  032154  005037  033130                    CLR     @#CORFLG        ;RESET INTERRUPT FLAG
6724  032160  104252                            ERROR   252             ;REPORT FAILURE. NO INTERRUPT.
6725  032162  000401                            BR      11$
6726  032164  104254                            ERROR   254
6727  032166                            11$:
6728
6729                                    ;TEST ADDD MODE 2
6730  032166                            COR5:
6731  032166  104413                            LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6732  032170  004737  032646                    JSR     PC,@#CORSUB
6733  032174  040202  111333  052525  1$:      .WORD   40202,111333,52525,70707        ;AC0
6734  032202  070707
6735  032204  032174                    2$:     .WORD   1$              ;R0
6736  032206  000205                    3$:     205                     ;FPS
6737  032210  172020                    4$:     ADDD    (R0)+,AC0       ;TEST INSTRUCTION
6738  032212  000240                            NOP
6739  032214  005037  033130                    CLR     @#CORFLG        ;RESET THE INTERRUPT FLAG
6740  032220  104252                            ERROR   252             ;REPORT FAILURE. NO INTERRUPT.
6741  032222  000401                            BR      11$
6742  032224  104255                            ERROR   255             ;CORRECT FLOWS FAILED.
6743  032226                            11$:
6744
6745                                    ;TEST ADDD MODE 3
6746  032226                            COR6:
6747  032226  104413                            LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6748  032230  004737  032646                    JSR     PC,@#CORSUB
6749  032234  040203  071735  072746  1$:      .WORD   40203,71735,72746,1     ;AC0
6750  032242  000001
6751  032244  032270                    2$:     .WORD   10$             ;R0
6752  032246  000206                    3$:     206                     ;FPS
6753  032250  172030                    4$:     ADDD    @(R0)+,AC0      ;TEST INSTRUCTION
6754  032252  000240                            NOP
6755  032254  005037  033130                    CLR     @#CORFLG                ;RESET THE INTERRUPT FLAG
6756  032260  104252                            ERROR   252             ;REPORT FAILURE. NO INTERRUPT.
6757  032262  000403                            BR      11$
6758  032264  104256                    5$:     ERROR   256             ;CORRECT FLOWS FAILED.
6759  032266  000401                            BR      11$
6760  032270  032234                    10$:    .WORD   1$              ;USED FOR THE ADDRESSING OF THE OPERAND
6761                                                                    ;IN THIS MODE.
6762  032272                            11$:
6763                                    ;TEST ADDD MODE 4
6764  032272                            COR7:
6765  032272  104413                            LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6766  032274  004737  032646                    JSR     PC,@#CORSUB
```

```
6767  032300  040204  123456  070123  1$:     .WORD   40204,123456,70123,45671        ;ACC
6768  032306  045671
6769  032310  032310                  2$:     .WORD   1$+10           ;R0
6770  032312  000212                  3$:     212                     ;FPS
6771  032314  172040                  4$:     ADDD    -(R0),AC0       ;TEST INSTRUCTION
6772  032316  000240                          NOP
6773  032320  005037  033130                  CLR     @#CORFLG        ;RESET THE INTERRUPT FLAG
6774  032324  104252                          ERROR   252             ;REPORT FAILURE. NO INTERRUPT.
6775  032326  000401                          BR      11$
6776  032330  104257                          ERROR   257             ;CORRECT FLOWS FAILED
6777  032332                      11$:
6778
6779                              ;TEST ADDD MODE 5
6780  032332                      COR8:
6781  032332  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6782  032334  004737  032646                  JSR     PC,@#CORSUB
6783  032340  040205  076543  021076  1$:     .WORD   40205,76543,21076,54321 ;AC0
6784  032346  054321
6785  032350  032376                  2$:     .WORD   10$+2           ;R0
6786  032352  000213                  3$:     213                     ;FPS
6787  032354  172050                  4$:     ADDD    @-(R0),AC0      ;TEST INSTRUCTION
6788  032356  000240                          NOP
6789  032360  005037  033130                  CLR     @#CORFLG        ;RESET THE INTERRUPT FLAG
6790  032364  104252                          ERROR   252             ;REPORT ERROR. NO INTERRUPT.
6791  032366  000403                          BR      11$
6792  032370  104260                  5$:     ERROR   260             ;CORRECT FLOWS FAILED.
6793  032372  000401                          BR      11$
6794  032374  032340                  10$:    .WORD   1$
6795  032376                      11$:
6796
6797                              ;TEST ADDD MODE 6
6798  032376                      COR9:
6799  032376  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6800  032400  004737  032646                  JSR     PC,@#CORSUB
6801  032404  040206  034353  063730  1$:     .WORD   40206,34353,63730,31323 ;AC0
6802  032412  031323
6803  032414  032403                  2$:     .WORD   1$-1            ;R0
6804  032416  000214                  3$:     214                     ;FPS
6805  032420  172060  000001          4$:     ADDD    1(R0),AC0       ;TEST INSTRUCTION
6806  032424  005037  033130                  CLR     @#CORFLG
6807  032430  104252                          ERROR   252             ;REPORT FAILURE NO TRAP.
6808  032432  000401                          BR      11$
6809  032434  104261                  5$:     ERROR   261
6810  032436                      11$:
6811
6812                              ;TEST ADDD MODE 7
6813  032436                      COR10:
6814  032436  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6815  032440  004737  032646                  JSR     PC,@#CORSUB
6816  032444  040210  070107  062426  1$:     .WORD   40210,70107,62426,55555 ;AC0
6817  032452  055555
6818  032454  032477                  2$:     .WORD   10$-1           ;R0
6819  032456  000204                  3$:     204                     ;FPS
6820  032460  172070  000001          4$:     ADDD    @1(R0),AC0      ;TEST INSTRUCTION
6821  032464  005037  033130                  CLR     @#CORFLG
6822  032470  104252                          ERROR   252             ;REPORT FAILURE NO TRAP
```

```
6823  032472  000403                          BR      11$
6824  032474  104262                  5$:     ERROR   262             ;CORRECT FLOWS FAILED.
6825  032476  000401                          BR      11$
6826  032500  032444                  10$:    .WORD   1$
6827  032502                          11$:
6828
6829                                  ;TEST DIVD MODE 1
6830  032502                          COR11:
6831  032502  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6832  032504  004737  032646                  JSR     PC,@#CORSUB
6833  032510  040211  033445  056677  1$:     .WORD   40211,33445,56677,001122        ;AC0
6834  032516  001122
6835  032520  032510                  2$:     .WORD   1$              ;R0
6836  032522  000205                  3$:     20S                     ;FPS
6837  032524  174410                  4$:     DIVD    (R0),AC0        ;TEST INSTRUCTION
6838  032526  000240                          NOP
6839  032530  005037  033130                  CLR     @#CORFLG
6840  032534  104252                          ERROR   252             ;REPORT FAILURE, NO TRAP.
6841  032536  000401                          BR      11$
6842  032540  104263                  5$:     ERROR   263             ;CORRECT FLOWS FAILED.
6843  032542                          11$:
6844
6845                                  ;TEST MULD MODE 1
6846  032542                          COR12:
6847  032542  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6848  032544  004737  032646                  JSR     PC,@#CORSUB
6849  032550  040212  165411  046252  1$:     .WORD   40212,165411,46252,63650        ;AC0
6850  032556  063650
6851  032560  032550                  2$:     .WORD   1$              ;R0
6852  032562  000210                  3$:     .WORD   210             ;FPS
6853  032564  171010                  4$:     MULD    (R0),AC0        ;TEST INSTRUCTION
6854  032566  000240                          NOP
6855  032570  005037  033130                  CLR     @#CORFLG
6856  032574  104252                          ERROR   252             ;REPORT FAILURE, NO TRAP.
6857  032576  000401                          BR      11$
6858  032600  104264                  5$:     ERROR   264             ;CORRECT FLOWS FAILED.
6859  032602                          11$:
6860
6861                                  ;TEST MODD MODE 1
6862  032602                          COR13:
6863  032602  104413                          LPERR                   ;SET UP THE LOOP ON ERROR ADDRESS.
6864  032604  004737  032646                  JSR     PC,@#CORSUB
6865  032610  040213  045654  054542  1$:     .WORD   40213,45654,54542,171623        ;AC0
6866  032616  171623
6867  032620  032610                  2$:     .WORD   1$              ;R0
6868  032622  000412                  3$:     .WORD   412             ;FPS
6869  032624  171410                  4$:     MODD    (R0),AC0        ;TEST INSTRUCTION.
6870  032626  000240                          NOP
6871  032630  005037  033130                  CLR     @#CORFLG
6872  032634  104252                          ERROR   252             ;REPORT FAILURE NO TRAP.
6873  032636  000401                          BR      11$
6874  032640  104265                  5$:     ERROR   265             ;CORRECT FLOWS FAILED.
6875  032642  000137  033146          11$:    JMP     @#CORDONE       ;FINISHED TEST!
6876
6877                                  ;THIS SUBROUTINE, CORSUB, IS CALLED TO SET UP THE OPERANDS
6878                                  ;AND CHECK THE RESULTS IN THIS TEST. IT IS CALLED THUS:
```

```
                                ;
                                        JSR     PC,@#CORSUB
                                1$:     .WORD   X,X,X,X         ;AC0 OPERAND
                                2$:     .WORD   X               ;R0
                                3$:     .WORD   X               ;FPS
                                4$:     INST                    ;TEST INSTRUCTION TO BE
                                                                ;EXECUTED.
                                        ADR                     ;AN ADDRESS OFFSET FOR
                                                                ;CERTAIN MODES OR NOP.
                                        ERROR   252             ;NO TRAP ERROR.
                                        BR      11$
                                5$:     ERROR   N               ;CORRECT FLOWS FAILURE.
                                        BR      11$             ;OPTIONAL FOR CERTAIN MODES.
                                10$:    .WORD   ADDRESS         ;OPTIONAL FOR CERTAIN MODES.
                                11$:
                                ;
                                ;CORSUB WILL PICK UP A POINTER TO THE ARGUMENTS, IN R1. AC0, R0 AND
                                ;THE FPS WILL BE SET TO THE DESIGNATED VALUES. THEN THE TEST MODULE
                                ;WILL BE SET UP TO INTERRUPT AND THE INSTRUCTION AT 4$ EXECUTED. IF
                                ;NO TRAP OCCURS THEN THE TEST MODULE IS FAULTY. WHEN THE TRAP OCCURS
                                ;THE PC ON THE STACK SHOULD BE 4$, AND AC0, R0 AND THE FPS SHOULD NOT
                                ;HAVE BEEN MODIFIED. IF EVERYTHING IS CORRECT CORSUB WILL RETURN TO
                                ;5$ PLUS TWO. IF AN ERROR IS DETECTED THEN CORSUB WILL RETURN TO THE
                                ;ERROR REPORT AT 5$.
                                ;NOTE THAT A FLAG, CORFLG, IS SET TO -1 WHEN AN INTERRUPT IS PENDING.
                                ;CORFLG IS ZERO OTHERWISE.
                                ;
032646  005037  033130        CORSUB: CLR     @#CORFLG        ;SET FLAG TO INDICATE NO INTERRUPT
                                                                ;PENDING.
032652  012601                  MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
032654  010102                  MOV     R1,R2           ;SET AC0.
032656  012700  000200          MOV     #200,R0
032662  170100                  LDFPS   R0
032664  172412                  LDD     (R2),AC0
032666  016100  000012          MOV     12(R1),R0       ;SET UP THE FPS.
032672  170100                  LDFPS   R0
032674  016100  000010          MOV     10(R1),R0       ;SET UP R0.
032700  010102                  MOV     R1,R2
032702  062702  000014          ADD     #14,R2
032706  010237  001236          MOV     R2,@#STMP2      ;SAVE ADDRESS OF INSTRUCTION IN CASE
                                                                ;OF ERROR.
032712  005037  177776          CLR     @#PSW           ;CLEAR THE PRIORITY TO ALLOW INTERRUPTS.
032716  012777  032744  000220  MOV     #CORTV,@CORTRP  ;SET UP THE INTERRUPT VECTOR.
032724  012767  177777  000176  MOV     #-1,CORFLG      ;SET THE FLAG TO INDICATE
                                                                ;AN INTERRUPT IS PENDING.
032732  012777  177777  000202  MOV     #-1,@CORINT     ;ENABLE THE TEST EQUIPMENT'S
                                                                ;TRAP FUNCTION AND GO
032740  000161  000014          JMP     14(R1)          ;EXECUTE THE INSTRUCTION.
                                ;
                                ;TRAP TO HERE WHEN THE INTERRUPT OCCURS.
032744  005137  033130        CORTV:  COM     @#CORFLG        ;FIRST SEE IF AN INTERRUPT WAS PENDING.
032750  001060                  BNE     CORTV1          ;IF NOT GO REPORT AN ERROR.
032752  012777  000000  000162  MOV     #0,@CORINT      ;MAKE SURE THE TEST EQUIPMENT
                                                                ;IS NOT INTERRUPT ENABLED.
                                ;
032760  170204                  STFPS   R4              ;GET THE FPS.
```

```
6935  032762  012702  000200                  MOV     #200,R2         ;GET ACO
6936  032766  170102                          LDFPS   R2
6937  032770  012702  033132                  MOV     #CORTMP,R2
6938  032774  174012                          STD     ACO,(R2)
6939  032776  012737  033132  001240          MOV     #CORTMP,@#$TMP3
6940  033004  010037  001244                  MOV     R0,@#$TMP5
6941  033010  010437  001250                  MOV     R4,@#$TMP7
6942  033014  011637  001254                  MOV     (SP),@#$TMP11
6943  033020  010102                          MOV     R1,R2
6944  033022  010237  001242                  MOV     R2,@#$TMP4
6945  033026  062702  000010                  ADD     #10,R2
6946  033032  012237  001246                  MOV     (R2)+,@#$TMP6
6947  033036  012237  001252                  MOV     (R2)+,@#$TMP10
6948  033042  010237  001256                  MOV     R2,@#$TMP12
6949  033046  021602                          CMP     (SP),R2         ;SEE IF THE TRAP OCCURRED
6950                                                                  ;AT THE CORRECT ADDRESS.
6951  033050  001016                          BNE     CORTVO
6952  033052  022626                          CMP     (SP)+,(SP)+     ;RESET THE STACK.
6953  033054  020061  000010                  CMP     R0,10(R1)       ;SEE IF R0 IS CORRECT.
6954  033060  001012                          BNE     CORTVO          ;BR IF NOT CORRECT.
6955  033062  010102                          MOV     R1,R2           ;SEE IF ACO WAS CORRECT
6956  033064  012703  033132                  MOV     #CORTMP,R3
6957  033070  012705  000004                  MOV     #4,R5
6958  033074  022223                  1$:     CMP     (R2)+,(R3)+
6959  033076  001003                          BNE     CORTVO          ;BRANCH IF INCORRECT.
6960  033100  077503                          SOB     R5,1$
6961  033102  000161  000032                  JMP     32(R1)          ;IF EVERYTHING IS CORRECT THEN RETURN.
6962
6963  033106  000161  000030          CORTVO: JMP     30(R1)          ;CORRECT FLOWS FAILED SO GO REPORT ERROR.
6964
6965  033112  011637  001236          CORTVI: MOV     (SP),@#$TMP2    ;AN INTERRUPT OCCURRED WHEN THE FLAG
6966                                                                  ;CORFLG, DID NOT INDICATE THAT ONE WAS
6967                                                                  ;PENDING SO REPORT SPURIOUS TRAP.
6968  033116  005037  033130                  CLR     @#CORFLG
6969  033122  022626                          CMP     (SP)+,(SP)+
6970  033124  104266                          ERROR   266
6971  033126  000407                          BR      CORDONE
6972
6973  033130  000000          CORFLG: .WORD   0
6974  033132  000000  000000  000000  CORTMP: .WORD   0,0,0,0
6975  033140  000000
6976
6977  033142  177774          CORINT: .WORD   177774          ;THIS IS THE ADDRESS, 177774, OF THE
6978                                                          ;TEST EQUIPMENT'S STATUS REGISTER.
6979                                                          ;THE CONTENTS OF CORINT CAN BE MODIFIED
6980                                                          ;IF THIS STATUS REGISTER'S ADDRESS IS
6981                                                          ;CHANGED.
6982  033144  000110          CORTRP: .WORD   110             ;THIS IS THE ADDRESS OF THE TEST EQUIPMENTS
6983                                                          ;TRAP VECTOR. LIKE THE STATUS REGISTER'S ADDRESS
6984                                                          ;DESCRIBED IMMEDIATELY ABOVE
6985                                                          ;THIS VECTOR CAN BE CHANGED, BUT THE
6986                                                          ;CONTENTS OF CORTRP MUST INDICATE THE
6987                                                          ;CHANGE.
6988
6989  033146          CORDONE:
699C  033146  104412                          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
```

```
                                                        ;SEE IF THE USER HAS EXPRESSED
                                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                                        ;VIRTUAL CONSOLE SWITCH REGISTER  (AE
                                                        ;THE USER TYPED CONTROL G?).
033150                          TST23:


                                .SBTTL  END OF PASS ROUTINE

                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                ;*IF SW12=1 INHIBIT TRACE TRAP
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO LOOP

033150                          $EOP:
                                        SCOPE
033150  000004                         CLR     $TSTNM          ;;ZERO THE TEST NUMBER
033152  005067  145724                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
033156  005067  146120                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
033162  005267  146136                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
033166  042767  100000  146130         DEC     (PC)+           ;;LOOP?
033174  005327                  $EOPCT: .WORD   1
033176  000001                          BGT     $DOAGN          ;;YES
033200  003074                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
033202  012737                  $ENDCT: .WORD   1
033204  000001                          $EOPCT
033206  033176                          TYPE    ,65$            ;;TYPE ASCIZ STRING
033210  104401  033216                  BR      64$             ;;GET OVER THE ASCIZ
033212  000407                  ;;65$:  .ASCIZ  <12><15>/END PASS #/
033214                          64$:
033234  033234  016746  146064         MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
033240                                                          ;;TYPE PASS NUMBER IN OCTAL
033240  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
033242     006                          .BYTE   6               ;;TYPE 6 DIGITS
033243     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
033244  104401  033252                  TYPE    ,67$            ;;TYPE ASCIZ STRING
033250  000421                          BR      66$             ;;GET OVER THE ASCIZ
033252                          ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
033314                          66$:
033314  016746  145572                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
033320                                                          ;;TOTAL NUMBER OF ERRORS IN OCTAL
033320  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
033322     006                          .BYTE   6               ;;TYPE 6 DIGITS
033323     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
033324  104401  001313                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
033330  005067  145556                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
033334  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
033340  001414                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
033342  005046                          CLR     -(SP)           ;;INSURE THE "T" BIT IS CLEAR
033344  012746  033352                  MOV     #$CLR.T,-(SP)   ;;SETUP FOR AN RTI OR RTT
```

```
                                END OF PASS ROUTINE

7047  033350  000426                      BR      $RTRN           ;;GO DO AN RTI OR RTT TO LOAD THE PSW
7048                                                              ;;WITH A CLEARED "T" BIT
7049  033352                    $CLR.T:
7050  033352  013700  000042              MOV     @#42,R0         ;;INSURE R0 CONTAINS THE MONITORS
7051  033356  001405                      BEQ     $DOAGN          ;;RETURN ADDRESS
7052  033360  000005                      RESET                   ;;CLEAR THE WORLD
7053  033362  004710            $ENDAD:   JSR     PC,(R0)         ;;GO TO MONITOR
7054  033364  000240                      NOP                     ;;SAVE ROOM
7055  033366  000240                      NOP                     ;;FOR
7056  033370  000240                      NOP                     ;;ACT11
7057  033372                    $DOAGN:
7058  033372  104400                      TRAP                    ;;PUSH OLD PSW AND PC ON STACK
7059  033374  042716  000020              BIC     #20,(SP)        ;;CLEAR THE "T" BIT
7060  033400  032777  010000  145532      BIT     #BIT12,@SWR     ;;RUN WITH TRACE TRAP?
7061  033406  001005                      BNE     1$              ;;BR IF NO
7062  033410  005167  000020              COM     $TBIT           ;;IS IT TIME FOR TRACE TRAP
7063  033414  100402                      BMI     1$              ;;BR IF NO
7064  033416  052716  000020              BIS     #20,(SP)        ;;SET TRACE TRAP
7065  033422  012746  033430    1$:       MOV     #$LOOP,-(SP)    ;;JUMP TO START OF TEST
7066  033426  000002            $RTRN:    RTI                     ;;RETURN--THIS IS CHANGED TO
7067                                                              ;;AN "RTT" IF "RTT" IS A LEGAL
7068                                                              ;;INSTRUCTION
7069  033430                    $LOOP:
7070  033430  000137                      JMP     @(PC)+          ;;RETURN
7071  033432  005034            $RTNAD:   .WORD   LOOP
7072  033434  000000            $TBIT:    .WORD   0               ;;"T" BIT STATE INDICATOR
7073  033436     377     377     000     $ENULL:   .BYTE   -1,-1,0       ;;NULL CHARACTER STRING
7074           033442                     .EVEN
7075
7076                                    .SBTTL  SCOPE HANDLER ROUTINE
7077
7078                                    ;;**************************************************************
7079                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7080                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7081                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7082                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7083                                    ;*SW14=1         LOOP ON TEST
7084                                    ;*SW11=1         INHIBIT ITERATIONS
7085                                    ;*SW09=1         LOOP ON ERROR
7086                                    ;*SW08=1         LOOP ON TEST IN SWR<7:0>
7087                                    ;*CALL
7088                                    ;*      SCOPE           ;;SCOPE=IOT
7089
7090  033442                    $SCOPE:
7091  033442  104406                      CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
7092  033444  032777  040000  145466  1$:       BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
7093  033452  001114                      BNE     $OVER           ;;YES IF SW14=1
7094                                    ;#####START OF CODE FOR THE XOR TESTER#####
7095  033454  000416            $XTSTR:   BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
7096                                                              ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7097  033456  013746  000004              MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7098  033462  012737  033502  000004      MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
7099  033470  005737  177060              TST     @#177060        ;;TIME OUT ON XOR?
7100  033474  012637  000004              MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
7101  033500  000463                      BR      $SVLAD          ;;GO TO THE NEXT TEST
7102  033502  022626            5$:       CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
```

```
 7103  033514  012637  000004          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 7104  033520  000423                  BR      7$              ;;LOOP ON THE PRESENT TEST
 7105  033522                  6$:;*****END OF CODE FOR THE XOR TESTER*****
 7106  033522  032777  000400  145420  BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
 7107  033530  001404                  BEQ     2$              ;;BR IF NO
 7108  033532  127767  145412  145352  CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
 7109  033540  001465                  BEQ     $OVER           ;;BR IF YES
 7110  033542  105767  145345      2$:  TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 7111  033546  001421                  BEQ     3$              ;;BR IF NO
 7112  033550  126767  145351  145335  CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 7113  033556  101015                  BHI     3$              ;;BR IF NO
 7114  033560  032777  001000  145362  BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 7115  033566  001404                  BEQ     4$              ;;BR IF NO
 7116  033570  016767  145324  145320  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 7117  033576  000446                  BR      $OVER
 7118  033600  105067  145307      4$:  CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 7119  033604  005067  145502          CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 7120  033610  000415                  BR      1$              ;;ESCAPE TO THE NEXT TEST
 7121  033612  032777  004000  145330  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
 7122  033620  001011                  BNE     1$              ;;BR IF YES
 7123  033622  005767  145506          TST     $PASS           ;;IF FIRST PASS OF PROGRAM
 7124  033626  001406                  BEQ     1$              ;;      INHIBIT ITERATIONS
 7125  033630  005267  145260          INC     $ICNT           ;;INCREMENT ITERATION COUNT
 7126  033634  026767  145452  145252  CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 7127  033642  002024                  BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 7128  033644  012767  000001  145242  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 7129  033652  016767  000052  145432          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 7130  033660  105267  145226      $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 7131  033664  116767  145222  145440          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 7132  033672  011667  145220              MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 7133  033676  011667  145216              MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 7134  033702  005067  145406              CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 7135  033706  112767  000001  145211      MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 7136  033714  016777  145172  145230  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 7137  033722  016716  145170              MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 7138  033726  000002                      RTI                     ;;FIXES PS
 7139  033730  000001          $MXCNT: 1                           ;;MAX. NUMBER OF ITERATIONS
 7140
 7141                          .SBTTL  ERROR HANDLER ROUTINE
 7142
 7143                          ;;**********************************************************************
 7144                          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 7145                          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 7146                          ;*AND GO TO ERTYPE ON ERROR
 7147                          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 7148                          ;*SW15=1        HALT ON ERROR
 7149                          ;*SW13=1        INHIBIT ERROR TYPEOUTS
 7150                          ;*SW10=1        BELL ON ERROR
 7151                          ;*SW09=1        LOOP ON ERROR
 7152                          ;*CALL
 7153                          ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 7154
 7155  033732                  $ERROR:
 7156  033732  104406                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 7157  033734  105267  145153      7$:  INCB    $ERFLG          ;;SET THE ERROR FLAG
 7158  033740  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
```

```
7159  033732  016777  145144  145202          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
7160  033740  032777  002000  145172          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
7161  033746  001402                          BEQ     1$              ;;NO - SKIP
7162  033750  104401  001306                  TYPE    ,$BELL          ;;RING BELL
7163  033754  005267  145132          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
7164  033760  011667  145132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
7165  033764  162767  000002  145124          SUB     #2,$ERRPC
7166  033772  117767  145120  145114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
7167  034000  032777  020000  145132          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
7168  034006  001004                          BNE     20$             ;;SKIP TYPEOUTS
7169  034010  004767  002124                  JSR     PC,ERTYPE       ;;GO TO USER ERROR ROUTINE
7170  034014  104401  001313                  TYPE    ,$CRLF
7171  034020                          20$:
7172  034020  122767  000001  145310          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
7173  034026  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
7174  034030  116767  145060  000004          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
7175  034036  004767  000740                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
7176  034042     000                  21$:    .BYTE   0
7177  034043     000                          .BYTE   0
7178  034044  000777                  22$:    BR      22$             ;;APT ERROR LOOP
7179  034046  005777  145066          2$:     TST     @SWR            ;;HALT ON ERROR
7180  034052  100002                          BPL     3$              ;;SKIP IF CONTINUE
7181  034054  000000                          HALT                    ;;HALT ON ERROR!
7182  034056  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
7183  034060  032777  001000  145052  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
7184  034066  001402                          BEQ     4$              ;;BR IF NO
7185  034070  016716  145014                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
7186  034074  005767  145204          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
7187  034100  001402                          BEQ     5$              ;;BR IF NONE
7188  034102  016716  145176                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
7189  034106                          5$:
7190  034106  022737  033362  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
7191  034114  001001                          BNE     6$              ;;BRANCH IF NO
7192  034116  000000                          HALT                    ;;YES
7193  034120                          6$:
7194  034120  032777  001000  145012          BIT     #BIT09,@SWR
7195  034126  001013                          BNE     ERM10
7196  034130  011637  001162                  MOV     (SP),@#$REG0            ;SEE IF ERROR #377
7197  034134  062737  177776  001162          ADD     #-2,@#$REG0
7198  034142  122777  000377  145012          CMPB    #377,@$REG0
7199  034150  001002                          BNE     ERM10
7200  034152  062716  000002                  ADD     #2,(SP)
7201  034156  000002                  ERM10:  RTI
7202
7203
7204                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
7205
7206                                  ;;**********************************************************************
7207                                  ;*SAVE R0-R5
7208                                  ;*CALL:
7209                                  ;*      SAVREG
7210                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
7211                                  ;*
7212                                  ;*TOP---(+16)
7213                                  ;* +2---(+18)
7214                                  ;* +4---R5
```

```
                                        ;* +6---R4
                                        ;* +8---R3
                                        ;*+10---R2
                                        ;*+12---R1
                                        ;*+14---R0
                                        
034160                                  $SAVREG:
034160  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
034162  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
034164  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
034166  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
034170  010446                                 MOV     R4,-(SP)        ;;PUSH R4 ON STACK
034172  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
034174  016646  000022                        MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
034200  016646  000022                        MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
034204  016646  000022                        MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
034210  016646  000022                        MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
034214  000002                                RTI
                                        
                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
034216                                  $RESREG:
034216  012666  000022                        MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
034222  012666  000022                        MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
034226  012666  000022                        MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
034232  012666  000022                        MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
034236  012605                                MOV     (SP)+,R5        ;;POP STACK INTO R5
034240  012604                                MOV     (SP)+,R4        ;;POP STACK INTO R4
034242  012603                                MOV     (SP)+,R3        ;;POP STACK INTO R3
034244  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
034246  012601                                MOV     (SP)+,R1        ;;POP STACK INTO R1
034250  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
034252  000002                                RTI
                                        
                                        .SBTTL  TYPE ROUTINE
                                        
                                        ;;*****************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*      TYPE
                                        ;*      MESADR
                                        ;*
                                        
034254  105767  144677                  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
034260  100002                                BPL     1$              ;;BR IF YES
034262  000000                                HALT                    ;;HALT HERE IF NO TERMINAL
034264  000430                                BR      3$              ;;LEAVE
```

```
7277  034266  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
7278  034270  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
7279  034274  122767  000001  145034            CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
7280  034302  001011                            BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
7281  034304  132767  000100  145025            BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
7282  034312  001405                            BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
7283  034314  010067  000004                    MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
7284  034320  004767  000446                    JSR     PC,$AT+3        ;;SPOOL MESSAGE TO APT
7285  034324  000000                    61$:    .WORD   0               ;;MESSAGE ADDRESS
7286  034326  132767  000340  145003    62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
7287  034334  001003                            BNE     60$             ;;YES,SKIP TYPE OUT
7288  034336  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
7289  034340  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
7290  034342  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
7291  034344  012600                    60$:    MOV     (SP)+,R0        ;;RESTORE R0
7292  034346  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
7293  034352  000002                            RTI                     ;;RETURN
7294  034354  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
7295  034360  001430                            BEQ     8$
7296  034362  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
7297  034366  001006                            BNE     5$
7298  034370  005726                            TST     (SP)+           ;;POP  <CR><LF> EQLIV
7299  034372  104401                            TYPE                    ;;TYPE A CR AND LF
7300  034374  001313                            $CRLF
7301  034376  105067  000130                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
7302  034402  000755                            BR      2$              ;;GET NEXT CHARACTER
7303  034404  004767  000056            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
7304  034410  126726  144542            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
7305  034414  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
7306  034416  016746  144532                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
7307                                            ;;AND THE NULL CHAR.
7308  034422  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
7309  034426  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
7310  034430  004767  000032                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
7311  034434  105367  000072                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
7312  034440  000770                            BR      7$              ;;LOOP
7313
7314                                    ;HORIZONTAL TAB PROCESSOR
7315
7316  034442  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
7317  034446  004767  000014            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
7318  034452  132767  000007  000052            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
7319  034460  001372                            BNE     9$              ;;TAB STOP
7320  034462  005726                            TST     (SP)+           ;;POP SPACE OFF STACK
7321  034464  000724                            BR      2$              ;;GET NEXT CHARACTER
7322  034466  105777  144456            $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
7323  034472  100375                            BPL     $TYPEC
7324  034474  116677  000002  144450            MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7325  034502  122766  000015  000002            CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
7326  034510  001003                            BNE     1$              ;;BRANCH IF NO
7327  034512  105067  000014                    CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
7328  034516  000406                            BR      $TYPEX          ;;EXIT
7329  034520  122766  000012  000002    1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
7330  034526  001402                            BEQ     $TYPEX          ;;BRANCH IF YES
7331  034530  105227                            INCB    (PC)+           ;;COUNT THE CHARACTER
7332  034532  000000            $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
```

```
                        TYPE ROUTINE

034534  000207          $TYPEX: RTS     PC


                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                        ;;**************************************************************
                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                        ;*      .BYTE   M               ;;M=1 OR 0
                        ;*                                      ;;1=TYPE LEADING ZEROS
                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                        ;*
                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                        ;*$TYPOS OR $TYPOC
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                        ;*
                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                        ;*
034536  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
034542  116667  000001  000211  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
034550  112667  000207          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
034554  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
034560  000406                  BR      $TYPON
034562  112767  000001  000171 $TYPOC: MOVB   #1,$OFILL      ;;SET THE ZERO FILL SWITCH
034570  112767  000006  000165        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
034576  112767  000005  000154 $TYPON: MOVB   #5,$OCNT       ;;SET THE ITERATION COUNT
034604  010346                  MOV     R3,-(SP)        ;;SAVE R3
034606  010446                  MOV     R4,-(SP)        ;;SAVE R4
034610  010546                  MOV     R5,-(SP)        ;;SAVE R5
034612  116704  000145          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
034616  005404                  NEG     R4
034620  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
034624  110467  000132          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
034630  116704  000125          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
034634  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
034640  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
034642  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
034644  000404                  BR      3$              ;;GO DO MSB
034646  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
034650  006105                  ROL     R5
034652  006105                  ROL     R5
034654  010503                  MOV     R5,R3
034656  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
034660  105367  000076          DECB    $OMODE          ;;TYPE THIS DIGIT?
034664  100016                  BPL     7$              ;;BR IF NO
034666  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
```

```
7383  034672  001002                        BNE     4$              ;;TEST FOR 0
7384  034674  005704                        TST     R4              ;;SUPPRESS THIS 0?
7385  034676  001403                        BEQ     5$              ;;BR IF YES
7386  034700  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
7387  034702  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
7388  034706  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
7389  034712  110367  000040                MOVB    R3,8$           ;;SAVE FOR TYPING
7390  034716  104401  034756                TYPE    ,8$             ;;GO TYPE THIS DIGIT
7391  034722  105367  000032        7$:     DECB    $OCNT           ;;COUNT BY 1
7392  034726  003347                        BGT     2$              ;;BR IF MORE TO DO
7393  034730  002402                        BLT     6$              ;;BR IF DONE
7394  034732  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
7395  034734  000744                        BR      2$              ;;GO DO THE LAST DIGIT
7396  034736  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
7397  034740  012604                        MOV     (SP)+,R4        ;;RESTORE R4
7398  034742  012603                        MOV     (SP)+,R3        ;;RESTORE R3
7399  034744  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
7400  034752  012616                        MOV     (SP)+,(SP)
7401  034754  000002                        RTI                     ;;RETURN
7402  034756  000                   8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
7403  034757  000                           .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
7404  034760  000                   $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
7405  034761  000                   $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
7406  034762  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
7407
7408                                .SBTTL  APT COMMUNICATIONS ROUTINE
7409
7410                                ;;**************************************************************
7411  034764  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
7412  034772  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
7413  035000  000403                        BR      $ATYC
7414  035002  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
7415  035010                        $ATYC:
7416  035010  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7417  035012  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7418  035014  105767  000206                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
7419  035020  001450                        BEQ     5$              ;;IF NOT:  BR
7420  035022  122767  000001  144306        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
7421  035030  001031                        BNE     3$              ;;IF NOT:  BR
7422  035032  132767  000100  144277        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
7423  035040  001425                        BEQ     3$              ;;IF NOT:  BR
7424  035042  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
7425  035046  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
7426  035054  005767  144236        1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
7427  035060  001375                        BNE     1$              ;;IF NOT:  WAIT
7428  035062  010067  144244                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
7429  035066  105720                2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
7430  035070  001376                        BNE     2$
7431  035072  166700  144234                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
7432  035076  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
7433  035100  010067  144230                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
7434  035104  012767  000004  144204        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
7435  035112  000413                        BR      5$
7436  035114  017667  000004  000016  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
7437  035122  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
7438  035130  016746  142642                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
```

```
7439  035134  004767  177114            JSR     PC,$TYPE        ;;CALL TYPE MACRO
7440  035140  000000            4$:     .WORD   0
7441  035142                    5$:
7442  035142  105767  000062    10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
7443  035146  001416                    BEQ     12$             ;;IF NOT:  BR
7444  035150  005767  144162            TST     $ENV            ;;RUNNING UNDER APT?
7445  035154  001413                    BEQ     12$             ;;IF NOT:  BR
7446  035156  005767  144134    11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
7447  035162  001375                    BNE     11$             ;;IF NOT:  WAIT
7448  035164  017667  000004  144126    MOV     @4(SP),$FATAL   ;;GET ERROR #
7449  035172  062766  000002  000004    ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
7450  035200  005267  144112            INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
7451  035204  105067  000020    12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
7452  035210  105067  000C13            CLRB    $LFLG           ;;CLEAR LOG FLAG
7453  035214  105067  000006            CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
7454  035220  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
7455  035222  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
7456  035224  000207                    RTS     PC              ;;RETURN
7457  035226     000    $MFLG:  .BYTE   0               ;;MESSG. FLAG
7458  035227     000    $LFLG:  .BYTE   0               ;;LOG FLAG
7459  035230     000    $FFLG:  .BYTE   0               ;;FATAL FLAG
7460          035232            .EVEN
7461          000200    APTSIZE=200
7462          000001    APTENV=001
7463          000100    APTSPOOL=100
7464          000040    APTCSUP=040
7465
7466                    .SBTTL  TTY INPUT ROUTINE
7467
7468                    ;;**********************************************************************
7469                    .ENABL  LSB
7470
7471                    ;;**********************************************************************
7472                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7473                    ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7474                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
7475                    ;*WHEN OPERATING IN TTY FLAG MODE.
7476  035232  022767  000176  143730  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
7477  035240  001074                    BNE     15$             ;;BRANCH IF NO
7478  035242  105777  143676            TSTB    @$TKS           ;;CHAR THERE?
7479  035246  100071                    BPL     15$             ;;IF NO, DON'T WAIT AROUND
7480  035250  117746  143672            MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
7481  035254  042716  177600            BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
7482  035260  022726  000007            CMP     #7,(SP)+        ;;IS IT A CONTROL G?
7483  035264  001062                    BNE     15$             ;;NO, RETURN TO USER
7484  035266  126727  143642  000001    CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
7485  035274  001456                    BEQ     15$             ;;BRANCH IF YES
7486
7487  035276  104401  035641            TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
7488  035302  104401  035646    $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
7489  035306  016746  142664            MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
7490  035312  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7491  035314  104401  035657            TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
7492  035320  005046            19$:    CLR     -(SP)           ;;CLEAR COUNTER
7493  035322  005046                    CLR     -(SP)           ;;THE NEW SWR
7494  035324  105777  143614    7$:     TSTB    @$TKS           ;;CHAR THERE?
```

```
7495  035330  100375                          BPL     7$              ;;IF NOT TRY AGAIN
7496
7497  035332  117746  143610                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
7498  035336  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
7499
7500
7501
7502  035342  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
7503  035346  001005                          BNE     10$             ;;BRANCH IF NOT
7504  035350  104401  035634                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U
7505  035354  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
7506  035360  000757                          BR      19$             ;;LET'S TRY IT AGAIN
7507
7508
7509  035362  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
7510  035366  001022                          BNE     16$             ;;BRANCH IF NO
7511  035370  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
7512  035374  001403                          BEQ     11$             ;;BRANCH IF YES
7513  035376  016677  000002  143534          MOV     2(SP),@SWR      ;;SAVE NEW SWR
7514  035404  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
7515  035410  104401  001313          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
7516  035414  126727  143515  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
7517  035422  001003                          BNE     15$             ;;BRANCH IF NOT
7518  035424  012777  000100  143512          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
7519  035432  000002          15$:    RTI                             ;;RETURN
7520  035434  004767  177026          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
7521  035440  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
7522  035444  002420                          BLT     18$             ;;BRANCH IF YES
7523  035446  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
7524  035452  003015                          BGT     18$             ;;BRANCH IF YES
7525  035454  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
7526  035460  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
7527  035464  001403                          BEQ     17$             ;;BRANCH IF YES
7528  035466  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
7529  035470  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
7530  035472  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
7531  035474  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
7532  035500  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
7533  035504  000707                          BR      7$              ;;GET THE NEXT ONE
7534  035506  104401  001312          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
7535  035512  000720                          BR      20$             ;;SIMULATE CONTROL-U
7536                                .DSABL  LSB
7537
7538
7539                                ;;**************************************************************
7540                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
7541                                ;*CALL:
7542                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
7543                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
7544                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
7545                                ;
7546
7547  035514  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
7548  035516  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
7549  035524  105777  143414          1$:     TSTB    @$TKS           ;;WAIT FOR
7550  035530  100375                          BPL     1$              ;;A CHARACTER
```

```
                                                ;;READ THE TTY
                                                ;;GET RID OF JUNK IF ANY
                                                ;;IS IT A CONTROL-S?
                                                ;;BRANCH IF NO
                                                ;;WAIT FOR A CHARACTER
                                                ;;LOOP UNTIL ITS THERE
                                                ;;GET CHARACTER
                                                ;;MAKE IT 7-BIT ASCII
                                                ;;IS IT A CONTROL-Q?
                                                ;;IF NOT DISCARD IT
                                                ;;YES, RESUME
                                                ;;IS IT UPPER CASE?
                                                ;;BRANCH IF YES
                                                ;;IS IT A SPECIAL CHAR?
                                                ;;BRANCH IF YES
                                                ;;MAKE IT UPPER CASE
                                                ;;GO BACK TO USER
                                                ;;CONTROL "U"
                                                ;;CONTROL "G"
                                                .SBTTL  TRAP DECODER

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

                                                ;;SAVE R0
                                                ;;GET TRAP ADDRESS
                                                ;;BACKUP BY 2
                                                ;;GET RIGHT BYTE OF TRAP
                                                ;;POSITION FOR INDEXING
                                                ;;INDEX TO TABLE
                                                ;;GO TO ROUTINE


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

                                                ;;MOVE THE PC DOWN
                                                ;;MOVE THE PSW DOWN
                                                ;;RESTORE THE PSW

                                                .SBTTL  TRAP TABLE

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

                                                ;       ROUTINE
                                                ;       -------
                                                ;;CALL=TYPE     TRAP+1(;104401)  TTY TYPEOUT ROUTINE
```

```
7607  035730  034562                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER WITH LEADING ZER
7608  035732  034536                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZER
7609  035734  034576                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER  AS PER LAST CALL
7610
7611  035736  035302                        $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
7612
7613  035740  035232                        $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
7614  035742  035514                        $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
7615  035744  034160                        $SAVREG ;;CALL=SAVREG   TRAP+10(104410) SAVE R0-R5 ROUTINE
7616  035746  034216                        $RESREG ;;CALL=RESREG   TRAP+11(104411) RESTORE R0-R5 ROUTINE
7617  035750  036710                        .RSET   ;;CALL=RSETUP   TRAP+12(104412) ROUTINE TO INITIALIZE AFTER EVERY TEST
7618  035752  036702                        .LPER   ;;CALL=LPERR    TRAP+13(104413) ROUTINE TO SET LOOP ON ERROR ADDRESS
7619           000030              $TERM=.-$TRPAD
7620
7621                               .SBTTL  POWER DOWN AND UP ROUTINES
7622
7623                               ;;**********************************************************************
7624                               ;POWER DOWN ROUTINE
7625  035754  012737  036132  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
7626  035762  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
7627  035770  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7628  035772  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7629  035774  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7630  035776  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7631  036000  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
7632  036002  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7633  036004  017746  143130                MOV     @$WR,-(SP)      ;;PUSH @SWR ON STACK
7634  036010  010667  000122                MOV     SP,$SAVR6       ;;SAVE SP
7635  036014  012737  036026  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
7636  036022  000000                        HALT
7637  036024  000776                        BR      .-2             ;;HANG UP
7638
7639                               ;;**********************************************************************
7640                               ;POWER UP ROUTINE
7641  036026  012737  036132  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
7642  036034  016706  000076                MOV     $SAVR6,SP       ;;GET SP
7643  036040  005067  000072                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
7644  036044  005267  000066          1$:   INC     $SAVR6          ;;WAIT FOR THE INC
7645  036050  001375                        BNE     1$              ;;OF  WORD
7646  036052  012677  143062                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
7647  036056  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
7648  036060  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
7649  036062  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
7650  036064  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
7651  036066  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
7652  036070  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
7653  036072  012737  035754  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
7654  036100  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
7655  036106  104401                        TYPE                    ;;REPORT THE POWER FAILURE
7656  036110  037104              $PWRMG: .WORD   POWERM          ;;POWER FAIL MESSAGE POINTER
7657  036112  012716                        MOV     (PC)+,(SP)      ;;RESTART AT START
7658  036114  004336              $PWRAD: .WORD   START           ;;RESTART ADDRESS
7659  036116  042766  000020  000002        BIC     #20,2(SP)       ;;CLEAR "T" BIT
7660  036124  005067  175304                CLR     $TBIT           ;;CLEAR THE "T" BIT FLAG
7661  036130  000002                        RTI
7662  036132  000000              $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
```

```
7663  036134  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
7664  036136  000000          $SAVR6: 0                       ;;PUT THE SP HERE
7665
7666
7667                                  .SBTTL  ERROR TYPE OUT ROUTINE
7668                          ;;**************************************************************
7669                          ;;**************************************************************
7670                          ;*THIS ROUTINE IS CALLED TO TYPE AN ERROR MESSAGE WHICH IS INCLUDED
7671                          ;*IN THE ERROR MESSAGE DATA TABLE. IT IS CALLED BY THE $ERROR ROUTINE
7672                          ;*OR BY FIRST SETTING $ITEMB EQUAL TO THE ERROR TABLE ITEM TO BE PRINTED
7673                          ;*OUT AND THEN EXECUTING A:
7674                          ;*              JSR     PC,ERTYPE
7675                          ;*
7676  036140  104401          ERTYPE: TYPE                                    ;TYPE A CRLF
7677  036142  001313                  .WORD   $CRLF
7678  036144  113737  001102  001232          MOVB    @#$TSTNM,@#$TMP0
7679  036152  042737  177400  001232          BIC     #177400,@#$TMP0
7680  036160  013737  001116  001234          MOV     @#$ERRPC,@#$TMP1        ;GET PC OF CALL
7681  036166  010046                          MOV     R0,-(SP)                ;SAVE R0
7682
7683  036170  113700  001114                  MOVB    @#$ITEMB,R0             ;GET THE ITEM NUMBER.
7684  036174  042700  177400                  BIC     #177400,R0
7685  036200  001005                          BNE     1$
7686
7687  036202  013746  001116                  MOV     @#$ERRPC,-(SP)          ;IF ZERO THEN JUST
7688  036206  104402                          TYPOC                           ;PRINT THE PC
7689  036210  000137  036610                  JMP     @#ERTS
7690
7691  036214  022700  000377          1$:     CMP     #377,R0
7692  036220  001005                          BNE     20$
7693  036222  016600  000004                  MOV     4(SP),R0
7694  036226  011000                          MOV     (R0),R0
7695  036230  062700  000400                  ADD     #400,R0
7696  036234  005300                  20$:    DEC     R0                      ;OTHERWISE MAKE RC AN
7697  036236  006300                          ASL     R0                      ;INDEX FOR THE TABLE.
7698  036240  006300                          ASL     R0
7699  036242  006300                          ASL     R0
7700  036244  062700  001442                  ADD     #$ERRTB,R0
7701
7702  036250  012037  036260                  MOV     (R0)+,@#2$              ;PICK UP THE ADDRESS
7703  036254  001404                          BEQ     3$                      ;OF THE EM, ERROR MESSAGE
7704  036256  104401                          TYPE
7705  036260  000000          2$:     .WORD   0
7706  036262  104401                  TYPE
7707  036264  001313                  .WORD   $CRLF
7708
7709  036266  012037  036276          3$:     MOV     (R0)+,@#4$              ;GET THE DH,DATA HEADER
7710  036272  001404                          BEQ     5$
7711  036274  104401                          TYPE
7712  036276  000000          4$:     .WORD   0
7713  036300  104401                  TYPE
7714  036302  001313                  .WORD   $CRLF
7715
7716  036304  010146          5$:     MOV     R1,-(SP)                ;SAVE R1,R2 AND R3
7717  036306  010246                  MOV     R2,-(SP)
7718  036310  010346                  MOV     R3,-(SP)
```

```
 7719
 7720  036312  012001                        MOV     (R0)+,R1                ;GET THE ADDRESS OF THE
 7721                                                                          ;DATA TABLE.
 7722  036314  001002                        BNE     6$
 7723  036316  000137  036576                JMP     @#ERT4                  ;RETURN IF NO DATA.
 7724
 7725  036322  011000              6$:       MOV     (R0),R0                 ;GET A POINTER TO THE DATA
 7726                                                                          ;FORMAT TABLE.
 7727  036324  105710              ERT1:     TSTB    (R0)                    ;FORMAT ZERO?
 7728  036326  001004                        BNE     7$
 7729
 7730  036330  013146                        MOV     @(R1)+,-(SP)            ;FORMAT ZERO SO TYPE
 7731  036332  104402                        TYPOC                           ;AN OCTAL NUMBER.
 7732  036334  000137  036560                JMP     @#ERT2
 7733
 7734  036340                      7$:
 7735  036340  122710  000002      8$:       CMPB    #2,(R0)                 ;FORMAT TWO?
 7736  036344  001011                        BNE     9$
 7737
 7738  036346  013102                        MOV     @(R1)+,R2               ;FORMAT TWO SO TYPE TWO
 7739  036350  012246                        MOV     (R2)+,-(SP)             ;OCTAL NUMBERS.
 7740  036352  104402                        TYPOC
 7741  036354  104401                        TYPE
 7742  036356  037153                        .WORD   SPACE
 7743  036360  011246                        MOV     (R2),-(SP)
 7744  036362  104402                        TYPOC
 7745  036364  000137  036560                JMP     @#ERT2
 7746
 7747  036370  122710  000003      9$:       CMPB    #3,(R0)                 ;FORMAT THREE?
 7748  036374  001021                        BNE     10$
 7749
 7750  036376  013102                        MOV     @(R1)+,R2               ;FORMAT THREE SO TYPE
 7751  036400  012246                        MOV     (R2)+,-(SP)             ;FOUR OCTAL NUMBERS.
 7752  036402  104402                        TYPOC
 7753  036404  104401                        TYPE
 7754  036406  037153                        .WORD   SPACE
 7755  036410  012246                        MOV     (R2)+,-(SP)
 7756  036412  104402                        TYPOC
 7757  036414  104401                        TYPE
 7758  036416  037153                        .WORD   SPACE
 7759  036420  012246                        MOV     (R2)+,-(SP)
 7760  036422  104402                        TYPOC
 7761  036424  104401                        TYPE
 7762  036426  037153                        .WORD   SPACE
 7763  036430  011246                        MOV     (R2),-(SP)
 7764  036432  104402                        TYPOC
 7765  036434  000137  036560                JMP     @#ERT2
 7766
 7767  036440  122710  000004      10$:      CMPB    #4,(R0)                 ;FORMAT FOUR?
 7768  036444  001005                        BNE     11$
 7769
 7770  036446  013146                        MOV     @(R1)+,-(SP)            ;FORMAR FOUR SO TYPE
 7771  036450  104403                        TYPOS                           ;AN OCTAL NUMBER
 7772  036452     016                        .BYTE   16                      ;SUPPRESSING LEADING ZEROES.
 7773  036453     000                        .BYTE   0
 7774  036454  000137  036560                JMP     @#ERT2
```

```
7775
7776  036460  122710  000005        11$:    CMPB    #5,(R0)                 ;FORMAT FIVE?
7777  036464  001006                        BNE     13$
7778
7779  036466  012137  036474                MOV     (R1)+,@#12$             ;FORMAT FIVE SO TYPE AN
7780  036472  104401                        TYPE                            ;ASCIZ STRING.
7781  036474  000000        12$:    .WORD   0
7782  036476  000137  036564                JMP     @#ERT3
7783
7784  036502  122710  000011        13$:    CMPB    #11,(R0)                ;FORMAT ELEVEN?
7785  036506  001006                        BNE     15$
7786
7787  036510  013137  036516                MOV     @(R1)+,@#14$            ;FORMAT ELEVEN SO PICK
7788  036514  104401                        TYPE                            ;A POINTER TO AN ASCIZ
7789  036516  000000        14$:    .WORD   0                               ;STRING.
7790  036520  000137  036564                JMP     @#ERT3
7791
7792  036524  122710  000012        15$:    CMPB    #12,(R0)                ;FORMAT TWELVE?
7793  036530  001012                        BNE     17$
7794
7795  036532  013102                        MOV     @(R1)+,R2               ;FORMAT TWELVE SO TYPE
7796  036534  012703  000006                MOV     #6,R3                   ;TYPE SIX OCTAL NUMBERS
7797  036540  012246        16$:    MOV     (R2)+,-(SP)
7798  036542  104402                        TYPOC
7799  036544  104401                        TYPE
7800  036546  037153                        .WORD   SPACE
7801  036550  077305                        SOB     R3,16$
7802  036552  000137  036560                JMP     @#ERT2
7803
7804  036556  000000        17$:    HALT                                    ;UNDEFINED FORMAT FOR DATA?????
7805
7806  036560  104401        ERT2:   TYPE                                    ;PRINT A TAB AFTER TYPING
7807  036562  037151                .WORD   $TAB                            ;AN DATA TABLE ENTRY
7808                                                                        ;OF ALL FORMATS EXCEPT
7809                                                                        ;ASCIZ, FORMATS 5 OR 11
7810
7811  036564  005200        ERT3:   INC     R0                              ;POINT TO THE NEXT FORMAT
7812  036566  005711                TST     (R1)                            ;END OF DATA TABLE.
7813  036570  001402                BEQ     ERT4
7814  036572  000137  036324                JMP     @#ERT1
7815
7816  036576  104401        ERT4:   TYPE                                    ;DONE.
7817  036600  001313                .WORD   $CRLF
7818  036602  012603                MOV     (SP)+,R3                        ;RESTORE R1,R2 AND R3
7819  036604  012602                MOV     (SP)+,R2
7820  036606  012601                MOV     (SP)+,R1
7821  036610  012600        ERT5:   MOV     (SP)+,R0                        ;RESTORE R0.
7822  036612  000207                RTS     PC                              ;AND RETURN.
7823
7824
7825
7826
7827                                .SBTTL  FPP SPURIOUS TRAP TO 244 HANDLER
7828                        ;;**************************************************************
7829                        ;;**************************************************************
7830                        ;THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.
```

```
                                        ;*THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
                                        ;*THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.
                                        ;*
036614  011637  001236          FPSPUR: MOV     (SP),@#$TMP2            ;SAVE PC OF TRAP.
036620  022626                          CMP     (SP)+,(SP)+             ;RESTORE SP.
036622  170200                          STFPS   R0                      ;GET FPS
036624  010037  001240                  MOV     R0,@#$TMP3
036630  170300                          STST    R0                      ;GET FEC
036632  010037  001242                  MOV     R0,@#$TMP4
036636  104247                  1$:     ERROR   247
036640  104412                          RSETUP                          ;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?).
036642  000137  033150                  JMP     @#$EOP

                                        .SBTTL  CPU SPURIOUS TRAP TO 4 HANDLER
                                ;;**************************************************************
                                ;;**************************************************************
                                ;*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.
                                ;*
036646  011637  001236          CPSPUR: MOV     (SP),@#$TMP2            ;SAVE PC OF TRAP.
036652  022626                          CMP     (SP)+,(SP)+
036654  104250                  1$:     ERROR   250
036656  104412                          RSETUP                          ;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?).
036660  000137  033150                  JMP     @#$ECP

                                        .SBTTL  CPU SPURIOUS TRAP TO 10 HANDLER
                                ;;**************************************************************
                                ;;**************************************************************
                                ;*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.
                                ;*
036664  011637  001236          CPTWO:  MOV     (SP),@#$TMP2            ;SAVE PC OF TRAP.
036670  022626                          CMP     (SP)+,(SP)+
036672  104251                  1$:     ERROR   251
036674  104412                          RSETUP                          ;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?).
036676  000137  033150                  JMP     @#$EOP


                                        .SBTTL  SET LOOP ON ERROR ADDRESS ROUTINE
                                ;;**************************************************************
                                ;;**************************************************************
```

```
7887                                    ;*
7888  036702  011637  001110            .LPER:  MOV     (SP),@#$LPERR
7889  036706  000002                            RTI
7890
7891                                            .SBTTL  FLAG RESET AND CONSOLE TEST ROUTINE
7892                                    ;;**********************************************************************
7893                                    ;;**********************************************************************
7894                                    ;*THIS ROUTINE WILL BE CALLED AT THE END OF EACH TEST TO
7895                                    ;*RESET THE STACK, CLEAR THE FPS AND SEE IF THE USER HAS TYPED
7896                                    ;* CONTROL G ON THE TERMINAL. IF THE USER HAS TYPED CONTROL G AND
7897                                    ;*THERE IS NO PHYSICAL CONSOLE SWITCH REGISTER THEN THE CONTENTS
7898                                    ;*OF THE SOFTWARE SWITCH REGISTER WILL BE TYPED IN OCTAL ON THE
7899                                    ;*TELETYPE AND THE USER CAN MODIFY IT.
7900                                    ;*
7901  036710  023727  001140  177570    .RSET:  CMP     @#SWR,#177570           ;SEE IF THERE IS A PHYSICAL
7902                                                                            ;CONSOLE SWITCH REGISTER.
7903  036716  001001                            BNE     1$                      ;BRANCH IF NO.
7904  036720  104406                            CKSWR                           ;OTHERWISE TYPE THE CONTENTS
7905                                                                            ;OF THE PROGRAM VIRTUAL SWITCH REGISTER
7906                                                                            ;AND GIVE THE USER A CHANCE TO
7907                                                                            ;MODIFY IT.
7908  036722  012737  036614  000244    1$:     MOV     #FPSPUR,@#FPVECT
7909  036730  012737  036646  000004            MOV     #CPSPUR,@#ERRVECT
7910  036736  012737  036664  000010            MOV     #CPTWO,@#10
7911  036744  011600                            MOV     (SP),R0                 ;SAVE RETURN ADDRESS.
7912  036746  012706  001100                    MOV     #STACK,SP               ;RESET THE STACK POINTER.
7913  036752  005004                            CLR     R4                      ;CLEAR THE FPS.
7914  036754  170104                            LDFPS   R4
7915  036756  000110                            JMP     (R0)                    ;RETURN.
7916
7917
7918                                            .NLIST  BFX

                                        ;THESE ARE SPECIAL MESSAGES:

036760  051124  050101  042520  MSA1:   .ASCIZ  'TRAPPED AT:'<TAB><TAB>
036776  054105  042520  052103  MSA2:   .ASCIZ  'EXPECTED TRAP AT:'<TAB>
037021     107  052117  051040  MSA3:   .ASCIZ  'GOT R0:'<TAB><TAB>
037033     105  050130  041505  MSA4:   .ASCIZ  'EXPECTED R0:'<TAB>
037051     107  052117  040440  MSA5:   .ASCIZ  'GOT AC0:'<TAB><TAB>
037064  054105  042520  052103  MSA6:   .ASCIZ  'EXPECTED AC0:'<TAB><TAB>


037104  050200  053517  051105  POWERM: .ASCIZ  <CRLF>'POWER FAILURE. PROGRAM RESTARTING.'<CRLF>
037151     011     000  $TAB:   .ASCIZ  <TAB>
037153     040  000040  SPACE:  .ASCIZ  ' '
037156  041501  047440  042520  MS1:    .ASCIZ  'AC OPERAND:'<TAB><TAB>
037174  051506  041522  047440  MS2:    .ASCIZ  'FSRC OPERAND:'<TAB><TAB>
037214  041501  020060  042502  MS3:    .ASCIZ  'AC0 BEFORE EXECUTION:'<TAB>
037243     101  030103  040440  MS4:    .ASCIZ  'AC0 AFTER EXECUTION:'<TAB>
037271     105  050130  041505  MS5:    .ASCIZ  'EXPECTED RESULT:'<TAB>
037313     107  052117  051040  MS6:    .ASCIZ  'GOT RESULT:'<TAB><TAB>
037331     106  040522  052103  MS7:    .ASCIZ  'FRACTIONAL RESULT:'<TAB>
037355     111  052116  043505  MS10:   .ASCIZ  'INTEGER RESULT: '<TAB>
037377     105  050130  041505  MS11:   .ASCIZ  'EXPECTED FRACTION:'<TAB>
037423     105  050130  041505  MS12:   .ASCIZ  'EXPECTED INTEGER:'<TAB>
```

```
037446  047514  042101  042105  MS37:   .ASCIZ  'LOADED DATA: '
037464  042522  042101  042040  MS40:   .ASCIZ  'READ DATA: '
037500  054105  042520  052103  MS415:  .ASCIZ  'EXPECTED DATA: '
037520  040524  040524  044440  MS41:   .ASCIZ  'DATA IN (R0) FSRC: '
037544  040504  040524  044440  MS42:   .ASCIZ  'DATA IN ACC: '
037562  047507  020124  042522  MS43:   .ASCIZ  'GOT RESULT: '
037577     105  050130  041505  MS44:   .ASCIZ  'EXPECTED RESULT: '
037621     200  042524  052123  CORMES: .ASCIZ  <CRLF>'TEST 22, TESTING INTERRUPTS.'<CRLF>


                                        ;THESE ARE ERROR MESSAGES:
037660  050106  020123  040502  EM1:    .ASCIZ  'FPS BAD AFTER CMPD (R),A.'
037712  041501  020060  047515  EM2:    .ASCIZ  'ACO MODIFIED BY CMPD (R),A.'
037745                          EM3:
037746  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
037772  041050  052125  042440          .ASCIZ  '(BUT ENBT) STATE 225 WENT TO 475 INSTEAD OF 075.'
040053                          EM4:
040053     106  051520  041040          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040077     050  052502  020124          .ASCIZ  '(BUT ENBT) STATE 225 WENT TO 075 INSTEAD OF 475.'
040160                          EM5:
040160  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040204  041050  052125  042440          .ASCIZ  '(BUT ENBT) STATE 035 WENT TO 075 INSTEAD OF 475.'
040265                          EM6:
040265     106  051520  041040          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040311     050  052502  020124          .ASCIZ  '(BUT ENBT) STATE 035 WENT TO 475 INSTEAD OF 075.'
040372                          EM7:
040372  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040416  041050  052125  042440          .ASCIZ  '(BUT ENBT YB) STATE 777 SHOULD HAVE GONE TO 007.'
040477                          EM10:
040477     106  051520  041040          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040523     050  052502  020124          .ASCIZ  '(BUT ENBT YB) STATE 777 SHOULD HAVE GONE TO 405.'
040604                          EM11:
040604  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040630  041050  052125  047040          .ASCIZ  '(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 010.'
040713                          EM12:
040713     106  051520  041040          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
040737     050  052502  020124          .ASCIZ  '(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 110.'
041022                          EM13:
041022  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
041046  044504  047104  052047          .ASCIZ  /DIDN'T TAKE THE PATH: STATE 456, TO 012. TO 363 TO 120.
041136                          EM14:
041136  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
041162  041050  052125  054040          .ASCIZ  '(BUT XNBT XZBT) STATE 363 WENT TO 140 INSTEAD OF 100.'
041250                          EM15:
041250  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
041274  041050  052125  054040          .ASCIZ  '(BUT XNBT XZBT) STATE 363 WENT TO 100 INSTEAD OF 140.'
041362                          EM16:
041362  050106  020123  040502          .ASCII  'FPS BAD AFTER CMPD.'<CRLF>
041406  044504  047104  052047          .ASCIZ  /DIDN'T TAKE THE PATH: STATE 777, TO 407./
041457     104  053111  020104  EM17:   .ASCIZ  'DIVD (R),A TRAPPED TO 244, FSRC=0 AND FID=1.'
041534  050106  020123  040502  EM20:   .ASCIZ  'FPS BAD AFTER DIVD (R),A.'
041566  042506  020103  040502  EM21:   .ASCIZ  'FEC BAD AFTER DIVD (R),A.'
041620  044504  042126  024040  EM22:   .ASCIZ  /DIVD (R),A DIDN'T TRAP TO 244. FSRC=0 AND FID=0./
041701     104  053111  020106  EM23:   .ASCIZ  'DIVF (R),A FAILED.'
041724  050106  020123  040502  EM32:   .ASCIZ  'FPS BAD AFTER DIVF (R),A.'
041756                          EM24:
```

```
                042054                  EM27=EM25
                                        EM30:
```

```
                                        EM31:
```

```
                                        EM33:   .ASCIZ  'DIVD (R),A FAILED.'
                                        EM34:   .ASCIZ  'FPS BAD AFTER DIVD (R),A.'
                                        EM35:
```

```
                                        EM36:
```



```
                                        EM37:   .ASCIZ  'MULF (R),A FAILED.'
                                        EM40:   .ASCIZ  'FPS BAD AFTER MULF (R),A.'
                                        EM41:
```

```
                                        EM42:
```

```
                                        EM43:
```



```
                                        EM44:
```



```
                                        EM45:
```

```
                                        EM46:   .ASCIZ  'FPS BAD AFTER MULD (R),A.'
                                        EM246:  .ASCIZ  'MULD (R),A FAILED.'
                                        EM47:
```

```
                                        EM50:
```

```
                                        EM51:
```

```
                                        EM52:
```

```
                043664  040502  020104  047503                  .ASCIZ  'BAD CONSTANT USED IN ROUNDING, FT=0.'
                043731     106  051520  041040  EM111:          .ASCIZ  'FPS BAD AFTER MULF (R),A. EXPECTED OVERFLOW.'
                044006  050106  020123  040502  EM112:          .ASCIZ  'FPS BAD AFTER MULF (R),A. EXPECTED UNDERFLOW.'
(0)             044064                          EM113:
(1)             044064  052515  043114  024040                  .ASCII  'MULF (R),A FAILED.'<CRLF>
                044107     105  050130  041505                  .ASCIZ  'EXPECTING OVERFLOW, FIV=0.'
(0)             044142                          EM114:
(1)             044142  052515  043114  024040                  .ASCII  'MULF (R),A FAILED.'<CRLF>
                044165     105  050130  041505                  .ASCIZ  'EXPECTING UNDERFLOW, FIU=0.'
                044221     115  046125  020106  EM115:          .ASCIZ  'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                044277     115  046125  020106  EM116:          .ASCIZ  'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
(0)             044356                          EM117:
(1)             044356  052515  043114  024040                  .ASCII  'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
                044434  041050  052125  043040                  .ASCIZ  '(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.'
(0)             044514                          EM120:
(1)             044514  052515  043114  024040                  .ASCII  'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
                044572  041050  052125  043040                  .ASCIZ  '(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'
(0)             044652                          EM121:
(1)             044652  052515  043114  024040                  .ASCII  'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                044727     050  052502  020124                  .ASCIZ  '(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'
(0)             045007                          EM122:
(1)             045007     115  046125  020106                  .ASCII  'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                045064  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'
                045144  050106  020123  040502  EM123:          .ASCIZ  'FPS BAD AFTER MULD (R),A. EXPECTING OVERFLOW.'
                045222  050106  020123  040502  EM124:          .ASCIZ  'FPS BAD AFTER MULD (R),A. EXPECTING UNDERFLOW.'
(0)             045301                          EM125:
(1)             045301     115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
                045324  054105  042520  052103                  .ASCIZ  'EXPECTING OVERFLOW, FIV=0.'
(0)             045357                          EM126:
(1)             045357     115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
                045402  054105  042520  052103                  .ASCIZ  'EXPECTING UNDERFLOW, FIU=0.'
                045436  052515  042114  024040  EM127:          .ASCIZ  'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                045514  052515  042114  024040  EM130:          .ASCIZ  'MULD (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
(0)             045573                          EM131:
(1)             045573     115  046125  020106                  .ASCII  'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
                045651     050  052502  020124                  .ASCIZ  '(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.'
(0)             045731                          EM132:
(1)             045731     115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
                045754  054105  042520  052103                  .ASCII  'EXPECTING UNDERFLOW, FIU=0.'
                046007     200  041050  052125                  .ASCIZ  <CRLF>'(BUT FD) STATE 115 WENT TO 424 INSTEAD OF 425.'
(0)             046067                          EM133:
(1)             046067     115  046125  020106                  .ASCII  'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
                046145     050  052502  020124                  .ASCIZ  '(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'
(0)             046225                          EM134:
(1)             046225     115  046125  020104                  .ASCII  'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                046302  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'
(0)             046362                          EM135:
(1)             046362  052515  042114  024040                  .ASCII  'MULD (R),A FAILED.'<CRLF>
                046405     105  050130  041505                  .ASCII  'EXPECTING OVERFLOW, FIV=0.'
                046437     200  041050  052125                  .ASCIZ  <CRLF>'(BUT FD) STATE 116 WENT TO 424 INSTEAD OF 425.'
(0)             046517                          EM136:
(1)             046517     115  046125  020104                  .ASCII  'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
                046574  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'
                046654  042506  020103  040502  EM137:          .ASCIZ  'FEC BAD AFTER MULF (R),A. EXPECTING OVERFLOW, FEC=10.'
                046742  042506  020103  040502  EM140:          .ASCIZ  'FEC BAD AFTER MULF (R),A. EXPECTING UNDERFLOW, FEC=12.'
                043731                          EM141=EM111
```

```
                044006                          EM142=EM112
                                                EM143:
 (0)  047231
 (1)  047031      115  046125  020106                  .ASCII  'MULF (R),A FAILED.'<CRLF>
      047054  054105  042520  052103                  .ASCIZ  'EXPECTING OVERFLOW, FIV=1.'
 (0)  047107                                    EM144:
 (1)  047107      115  046125  020106                  .ASCII  'MULF (R),A FAILED.'<CRLF>
      047132  054105  042520  052103                  .ASCIZ  'EXPECTING UNDERFLOW, FIU=1.'
 (0)  047166                                    EM145:
 (1)  047166  052515  043114  024040                  .ASCII  'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
      047253      050  052502  020124                  .ASCIZ  '(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'
 (0)  047333                                    EM146:
 (1)  047333      115  046125  020106                  .ASCII  'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
      047420  041050  052125  043040                  .ASCIZ  '(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'
 (0)  047500                                    EM147:
 (1)  047500  052515  043114  024040                  .ASCII  'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
      047564  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
 (0)  047644                                    EM150:
 (1)  047644  052515  043114  024040                  .ASCII  'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
      047730  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF A36.'
      050010  042506  020103  040502  EM151:          .ASCIZ  'FEC BAD AFTER MULD (R),A. EXPECTING OVERFLOW, FEC=10.'
      050076  042506  020103  040502  EM152:          .ASCIZ  'FEC BAD AFTER MULD (R),A. EXPECTING UNDERFLOW, FEC=12.'
              045144                            EM153=EM123
              045222                            EM154=EM124
 (0)  050165                                    EM155:
 (1)  050165      115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
      050210  054105  042520  052103                  .ASCIZ  'EXPECTING OVERFLOW, FIV=1.'
 (0)  050243                                    EM156:
 (1)  050243      115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
      050266  054105  042520  052103                  .ASCIZ  'EXPECTING UNDERFLOW, FIU=1.'
 (0)  050322                                    EM157:
 (1)  050322  052515  042114  024040                  .ASCII  'MULD (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
      050407      050  052502  020124                  .ASCIZ  '(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'
 (0)  050467                                    EM160:
 (1)  050467      115  046125  020104                  .ASCII  'MULD (R),A FAILED.'<CRLF>
      050512  054105  042520  052103                  .ASCII  'EXPECTING UNDERFLOW, FIU=1.'
      050545      200  041050  052125                  .ASCIZ  <CRLF>'(BUT FD) STATE 155 WENT TO 426 INSTEAD OF 427.'
 (0)  050625                                    EM161:
 (1)  050625      115  046125  020104                  .ASCII  'MULD (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
      050712  041050  052125  043040                  .ASCIZ  '(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'
 (0)  050772                                    EM162:
 (1)  050772  052515  042114  024040                  .ASCII  'MULD (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
      051055  041050  052125  043040                  .ASCIZ  '(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
 (0)  051136                                    EM163:
 (1)  051136  052515  042114  024040                  .ASCII  'MULD (R),A FAILED.'<CRLF>
      051161      105  050130  041505                  .ASCII  'EXPECTING OVERFLOW, FIV=1.'
      051213      200  041050  052125                  .ASCIZ  <CRLF>'(BUT FD) STATE 700 WENT TO 426 INSTEAD OF 427.'
 (0)  051273                                    EM164:
 (1)  051273      115  046125  020104                  .ASCII  'MULD (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
      051357      050  052502  020124                  .ASCIZ  '(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF 136.'
      051437      106  051520  041040  EM55:          .ASCIZ  'FPS BAD AFTER MODF (R),A.'
      051471      115  042117  020106  EM53:          .ASCIZ  'MODF (R),A FRACTION BAD.'
      051522  047515  043104  024040  EM54:          .ASCIZ  'MODF (R),A INTEGER BAD.'
 (0)  051552                                    EM56:
 (1)  051552  047515  043104  024040                  .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
      051603      101  030103  042040                  .ASCIZ  'ACO DID NOT GET 0 IN STATE 424.'
 (0)  051643                                    EM57:
```

```
 051643     115  042117  020106          .ASCII  'MODF (R),A INTEGER BAD.'<CRLF>
 051673     101  030503  042040          .ASCIZ  'AC1 DID NOT GET 0 IN STATE 142.'
 051733                              EM60:
 051733     115  042117  020106          .ASCII  'MODF (R),A INTEGER BAD.'<CRLF>
 051763     101  030503  042040          .ASCIZ  'AC1 DID NOT GET THE INTEGER IN STATE 134.'
 052035                              EM61:
 052035     115  042117  020106          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 052066  020101  040502  020104          .ASCII  'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
 052144  047600  020122  041050          .ASCIZ  <CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
 052231                              EM62:
 052231     115  042117  020106          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 052262  020101  040502  020104          .ASCII  'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
 052340  047600  020122  041050          .ASCIZ  <CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
 052425                              EM63:
 052425     115  042117  020106          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 052456  041050  052125  055040          .ASCIZ  '(BUT ZBT) STATE 532 WENT TO 102 INSTEAD OF 122.'
 052536                              EM64:
 052536  047515  043104  024040          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 052567     050  052502  020124          .ASCIZ  '(BUT ENBT EZBT) STATE 041 WENT TO 046 INSTEAD OF 245.'
 052655                              EM65:
 052655     115  042117  020106          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 052706  041050  052125  043040          .ASCIZ  '(BUT FT) STATE 126 SHOULD HAVE GONE TO 133. FT=0.'
 052770                              EM66:
 052770  047515  043104  024040          .ASCII  'MODF (R),A FRACTION BAD.'<CRLF>
 053021     123  043511  020116          .ASCIZ  'SIGN BIT BAD.'
 053037                              EM67:
 053037     115  042117  020106          .ASCII  'MODF (R),A INTEGER BAD.'<CRLF>
 053067     123  043511  020116          .ASCIZ  'SIGN BIT BAD IN STATE 733.'
 053122  047515  042104  024040  EM70:   .ASCIZ  'MODD (R),A FRACTION BAD.'
 053153     115  042117  020104  EM71:   .ASCIZ  'MODD (R),A INTEGER BAD.'
 053203     106  051520  041040  EM72:   .ASCIZ  'FPS BAD AFTER MODD (R),A.'
         053203                  EM73=EM72
 053235                              EM74:
 053235     115  042117  020104          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
 053265     050  052502  020124          .ASCIZ  '(BUT FD) STATE 231 WENT TO 142 INSTEAD OF 143.'
 053344                              EM75:
 053344  047515  042104  024040          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF>
 053375     101  030103  043440          .ASCIZ  'AC0 GETS 0 IN STATE 425 FAILED.'
 053435                              EM76:
 053435     115  042117  020104          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
 053465     101  030503  043440          .ASCIZ  'AC1 GETS 0 IN STATE 143 FAILED.'
 053525                              EM77:
 053525     115  042117  020104          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF>
 053556  041050  052125  043040          .ASCIZ  '(BUT FD) STATE 526 WENT TO 134 INSTEAD OF 135.'
 053635                              EM100:
 053635     115  042117  020104          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
 053665     123  043511  020116          .ASCIZ  'SIGN BIT BAD IN STATE 526.'
 053720                              EM101:
 053720  047515  042104  024040          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF>
 053751     101  041040  042101          .ASCII  'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046,'
 054027     200  051117  024040          .ASCIZ  <CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
 054114                              EM102:
 054114  047515  042104  024040          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF>
 054145     101  041040  042101          .ASCII  'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046,'
 054223     200  051117  024040          .ASCIZ  <CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
 054310                              EM103:
```

```
 1)  054310  047515  042104  024040          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF
     054341     050  052502  020124          .ASCIZ  '(BUT ZBIT) STATE 532 WENT TO 122 INSTEAD OF 103.'
 0)  054422                             EM104:
 1)  054422  047515  042104  024040          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
     054452  042523  020124  047111          .ASCII  'SET INTEGER IN AC1 FAILED,'
     054504  047600  020122  041050          .ASCIZ  <CRLF>'OR (BUT FD) STATE 733 WENT TO 156 INSTEAD OF 157.'
 0)  054567                             EM105:
 1)  054567     115  042117  020104          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
     054617     050  052502  020124          .ASCIZ  '(BUT FD) STATE 122 WENT TO 424 INSTEAD OF 425.'
 0)  054676                             EM106:
 1)  054676  047515  042104  024040          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
     054726  041050  052125  043040          .ASCIZ  '(BUT FD) STATE 246 WENT TO 126 INSTEAD OF 127.'
 0)  055005                             EM107:
 1)  055005     115  042117  020104          .ASCII  'MODD (R),A INTEGER BAD.'<CRLF>
     055036     050  052502  020124          .ASCIZ  '(BUT FD) STATE 446 WENT TO 126 INSTEAD OF 127.'
 0   055114                             EM110:
 1   055114  047515  042104  024040          .ASCII  'MODD (R),A FRACTION BAD.'<CRLF>
     055145     050  052502  020124          .ASCIZ  '(BUT FT) STATE 127 WENT TO 313 INSTEAD OF 113.'
 0)  055224                             EM165:
 1)  055224  047515  043104  024040          .ASCII  /MODF (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW.
     055307     000                          .BYTE   0
 0)  055310                             EM166:
 1)  055310  047515  043104  024040          .ASCII  /MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.
     055372     000                          .BYTE   0
 0)  055373                             EM167:
 1)  055373     106  051520  041040          .ASCII  /FPS BAD AFTER MODF (R),A./
     055424     000                          .BYTE   0
 0)  055425                             EM170:
 1)  055425     106  041505  041040          .ASCII  /FEC BAD AFTER MODF (R),A./
     055456  054105  042520  052103          .ASCIZ  'EXPECTING UNDERFLOW, FEC=12.'
 0)  055513                             EM171:
 1)  055513     115  042117  020106          .ASCII  /MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.
     055575     200  041501  020061          .ASCIZ  <CRLF>'AC1 GETS 0 IN STATE 126 FAILED.'
 0)  055636                             EM172:
 1)  055636  042506  020103  040502          .ASCII  /FEC BAD AFTER MODF (R),A./
     055667     105  050130  041505          .ASCIZ  'EXPECTING OVERFLOW, FEC=10.'
 0)  055723                             EM173:
 1)  055723     115  042117  020106          .ASCII  /MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
     056005     200  041050  052125          .ASCIZ  <CRLF>'(BUT FIV FD) STATE 520 WENT TO 142 INSTEAD OF 162.'
 0   056071                             EM174:
 1)  056071     115  042117  020106          .ASCII  /MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
     056153     200  041050  052125          .ASCIZ  <CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 142.'
 0)  056237                             EM175:
 1)  056237     115  042117  020106          .ASCII  /MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
     056321     200  044523  047107          .ASCIZ  <CRLF>'SIGN BAD IN STATE 517.'
 0)  056351                             EM176:
 1)  056351     115  042117  020104          .ASCII  /MODD (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW.
     056434     000                          .BYTE   0
 0)  056435                             EM177:
 1)  056435     115  042117  020104          .ASCII  /MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.
     056517     000                          .BYTE   0
 0)  056520                             EM200:
 1)  056520  050106  020123  040502          .ASCII  /FPS BAD AFTER MODD (R),A./
     056551     000                          .BYTE   0
 0   056552                             EM201:
 1   056552  042506  020103  040502          .ASCII  /FEC BAD AFTER MODD (R),A.
```

```
056603     200  054105  042520          .ASCIZ  <CRLF>'EXPECTING UNDERFLOW, FEC=12.'
056641                              EM202:
056641     115  042117  020104          .ASCII   MODD (R),A INTEGER BAD. RESULT OVER OR UNDEF FLOW.
056723     200  041050  052125          .ASCIZ  <CRLF>'(BUT FD) STATE 241 WENT TO 126 INSTEAD OF 127.'
057003                              EM203:
057003     115  042117  020104          .ASCII  /MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.
057065     200  041050  052125          .ASCIZ  <CRLF>'(BUT FD) STATE 047 WENT TO 126 INSTEAD OF 127.'
057145                              EM204:
057145     136  041505  041040          .ASCII  /FEC BAD AFTER MODD (R),A./
057176  042600  050130  041505          .ASCIZ  <CRLF>'EXPECTING OVERFLOW, FEC=10.'
057233                              EM205:
057233     115  042117  020104          .ASCII  /MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
057315     200  041050  052125          .ASCIZ  <CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 163.'
057401                              EM206:
057401     115  042117  020104          .ASCII  /MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
057463     200  041050  052125          .ASCIZ  <CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 143.'

                                        ;*
057547                              EM207:
057547     101  042104  020104          .ASCIZ  /ADDD (R),A PRODUCED A BAD RESULT./
057611                              EM210:
057611     124  042510  043040          .ASCIZ  /THE FPS WAS BAD AFTER ADDD (R),A./
057653     101  042104  020104  EM211:  .ASCII  'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
057726  053600  047105  020124          .ASCII  <CRLF>'WENT FROM STATE 663 TO 313,'<CRLF>
057763     111  051516  042524          .ASCIZ  'INSTEAD OF FROM 663 TO 353.'
060017                              EM212:
060017     101  042104  020104          .ASCII  'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060072  052200  042510  043040          .ASCII  <CRLF>'THE FPS WAS BAD.'<CRLF>
060114  044504  020104  047516          .ASCII  'DID NOT TAKE THE PATH:'<CRLF>
060143     106  047522  020115          .ASCIZ  /FROM STATE 664, TO 505, TO 251./
060203                              EM213:
060203     101  042104  020104          .ASCII  'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060256  052200  042510  043040          .ASCII  <CRLF>'THE FPS WAS BAD.'<CRLF>
060300  044504  020104  047516          .ASCII  'DID NOT TAKE THE PATH:'<CRLF>
060327     106  047522  020115          .ASCIZ  /FROM STATE 664, TO 505, TO 253./
060367                              EM214:
060367     101  042104  020104          .ASCII  'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060442  052200  042510  043040          .ASCII  <CRLF>'THE FPS WAS BAD.'<CRLF>
060464  044504  020104  047516          .ASCII  'DID NOT TAKE THE PATH:'<CRLF>
060513     106  047522  020115          .ASCIZ  /FROM STATE 664, TO 705, TO 735./
060553                              EM215:
060553     101  042104  020104          .ASCII  'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060626  052200  042510  043040          .ASCII  <CRLF>'THE FPS WAS BAD.'<CRLF>
060650  044504  020104  047516          .ASCII  'DID NOT TAKE THE PATH:'<CRLF>
060677     106  047522  020115          .ASCIZ  /FROM STATE 664, TO 705, TO 737./
060737     124  042510  024040  EM216:  .ASCII  'THE (BUT FIU FORK IN THE OVER\UNDER FLOWS FAILED. FIU =1.'
061030  053600  047105  020124          .ASCII  <CRLF>'WENT FROM STATE 331 TO 115.'<CRLF>
061065     111  051516  042524          .ASCIZ  'INSTEAD OF FROM 331 TO 155.'
061121     101  042104  020104  EM217:  .ASCIZ  'ADDD (R)A TRAPPED TO 244,. FID=1.'
061163                              EM220:
061163     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./<CRLF>
061216  044124  020105  042522          .ASCII  'THE RESULT WAS AN OVERFLOW CONDITION BUT FIV= 0.'
061276  024200  052502  020124          .ASCIZ  <CRLF>/(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.
061357     111  051516  042524          .ASCIZ  /INSTEAD OF FROM 133 TO 116./
061413                              EM221:
061413     101  042104  020104          .ASCII  /ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
```

```
(1)  061455     124  042510  051040          .ASCII  'THE RESULT WAS A OVERFLOW CONDITION AND FI.=. ' .R_F
(1)  061534  044124  020105  041050          .ASCII  /THE (BUT FIV) FORK FAILED./<CRLF>
(1)  061567     127  047105  020124          .ASCII  /WENT FROM STATE 133 TO 116,/<CRLF
(1)  061623     111  051516  042524          .ASCIZ  /INSTEAD OF FROM 133 TO 136./
(0)  061657                                 EM222:
(1)  061657     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./<CRLF>
(1)  061712  044124  020105  042522          .ASCII  'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU= 0.'
(1)  061773     200  041050  052125          .ASCIZ  <CRLF>/(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115./
(1)  062054  047111  052123  040505          .ASCIZ  /INSTEAD OF FROM 331 TO 115./
(0)  062110                                 EM223:
(1)  062110  042101  042104  024040          .ASCII  /ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
(1)  062152  044124  020105  042522          .ASCII  'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.' CRLF
(1)  062232  044124  020105  041050          .ASCII  /THE (BUT FIU) FORK FAILED./<CRLF>
(1)  062265     127  047105  020124          .ASCII  /WENT FROM STATE 331 TO 115,/<CRLF>
(1)  062321     111  051516  042524          .ASCIZ  /INSTEAD OF FROM 331 TO 155./
(0)  062355                                 EM224:
(1)  062355     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./<CRLF>
(1)  062410  044124  020105  042522          .ASCII  'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU= 0.'
(1)  062471     200  041050  052125          .ASCIZ  <CRLF>/(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115./
(1)  062552  047111  052123  040505          .ASCIZ  /INSTEAD OF FROM 137 TO 115./
(0)  062606                                 EM225:
(1)  062606  042101  042104  024040          .ASCII  /ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
(1)  062650  044124  020105  042522          .ASCII  'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.'<CRLF>
(1)  062730  044124  020105  041050          .ASCII  /THE (BUT FIU) FORK FAILED./<CRLF>
(1)  062763     127  047105  020124          .ASCII  /WENT FROM STATE 137 TO 115,/<CRLF>
(1)  063017     111  051516  042524          .ASCIZ  /INSTEAD OF FROM 137 TO 155./
(0)  063053                                 EM226:
(1)  063053     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./
(1)  063105     200  042502  040503          .ASCII  <CRLF>'BECAUSE OF AN EXPECTED OVERFLOW CONDITION,'<CRLF>
(1)  063161     102  052125  052040          .ASCIZ  'BUT THE FEC WAS BAD.'
(0)  063206                                 EM227:
(1)  063206  042101  042104  024040          .ASCII  /ADDD (R),A TRAPPED TO 244./
(1)  063240  041200  041505  052501          .ASCII  <CRLF>'BECAUSE OF AN EXPECTED OVERFLOW CONDITION,'<CRLF>
(1)  063314  052502  020124  044124          .ASCIZ  'BUT THE FPS WAS BAD.'
(0)  063341                                 EM230:
(1)  063341     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./
(1)  063373     200  042502  040503          .ASCII  <CRLF>'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION,'<CRLF>
(1)  063450  052502  020124  044124          .ASCIZ  'BUT THE FEC WAS BAD.'
(0)  063475                                 EM231:
(1)  063475     101  042104  020104          .ASCII  /ADDD (R),A TRAPPED TO 244./
(1)  063527     200  042502  040503          .ASCII  <CRLF>'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION,'<CRLF>
(1)  063604  052502  020124  044124          .ASCIZ  'BUT THE FPS WAS BAD.'
             057611                         EM232=EM210



(0)  063631                                 EM233:
(1)  063631     114  041504  042106          .ASCIZ  \LDCFD (R)+,A RESULT INCORRECT.\
(0)  063670                                 EM234:
(1)  063670  030122  041040  042101          .ASCII  \R0 BAD AFTER LDCFD (R)+,A.\
(1)  063722  040600  041040  042101          .ASCIZ  <CRLF>'A BAD CONSTANT WAS USED.'
 2.  063754                                 EM235:
```

```
                                                .ASCIZ  \PC BAD AFTER LDCFD #NUM,A. TRAP TO 4.
064022                                  EM236:
                                                .ASCIZ  \PC BAD AFTER LDCFD #NUM,A.\
064055                                  EM237:
064055  122  020060  040502                     .ASCII  \R0 BAD AFTER LDCDF (R)+,A.\
064107  200  020101  040502                     .ASCIZ  <CRLF>'A BAD CONSTANT WAS USED.'
064141                                  EM240:
                                                .ASCIZ  \FPS BAD AFTER LDCFD (R)+,A.\
064175                                  EM241:
064175  114  041504  042106                     .ASCII  \LDCFD (R)+,A FAILED.\
064221  200  044124  020105                     .ASCII  <CRLF>'THE FD '
064231  102  052111  053440                     .ASCII  'BIT WAS NOT COMPLIMENTED '
064262  047111  051440  040524                 .ASCIZ  'IN STATE 017.'
064300                                  EM242:
                                                .ASCIZ  \FPS BAD AFTER LDCDF (R)+,A.\
064334                                  EM243:
064334  042114  042103  020106                 .ASCII  \LDCDF (R)+,A FAILED.\
064360  052200  042510  043040                 .ASCII  <CRLF>'THE FD '
064370  044502  020124  040527                 .ASCII  'BIT WAS NOT COMPLIMENTED '
064421  111  020116  052123                     .ASCIZ  'IN STATE 017.'
064437                                  EM244:
064437  114  041504  043104                     .ASCIZ  \LDCDF (R)+,A RESULT INCORRECT.\

064476  042114  043103  020104  EM245:  .ASCII  'LDCFD (R),A FAILED.'
064521  200  042523  020124                     .ASCII  <CRLF>'SET SIGN FAILED '
064542  047111  051440  040524                 .ASCIZ  'IN STATE 512.'
064560  047125  054105  042520  EM247:  .ASCIZ  'UNEXPECTED FPP TRAP TO 244.'
064614  047125  054105  042520  EM250:  .ASCIZ  'UNEXPECTED CPU TRAP TO 4.'
064646  047125  054105  042520  EM251:  .ASCIZ  'UNEXPECTED CPU TRAP TO 10.'

064701  103  051117  042522  EM252:  .ASCIZ  'CORRECT FLOWS INTERRUPT TEST MODULE FAILED TO INTERRUPT '

064772                                  EM253:
064772  042101  042104  040440                 .ASCIZ  /ADDD AC0,AC0 FAILED IN THE INTERRUPT CORRECT FLOWS./
065056                                  EM254:
065056  042101  042104  024040                 .ASCIZ  /ADDD (R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS./
065143                                  EM255:
065143  101  042104  020104                     .ASCIZ  /ADDD (R0)+,AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065231                                  EM256:
065231  101  042104  020104                     .ASCIZ  /ADDD @(R0)+,AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065320                                  EM257:
065320  042101  042104  026440                 .ASCIZ  /ADDD -(R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065406                                  EM260:
065406  042101  042104  040040                 .ASCIZ  /ADDD @-(R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065475                                  EM261:
065475  101  042104  020104                     .ASCIZ  /ADDD NUM(R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065565                                  EM262:
065565  101  042104  020104                     .ASCIZ  /ADDD @NUM(R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065656                                  EM263:
065656  044504  042126  024040                 .ASCIZ  /DIVD (R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
065743                                  EM264:
065743  115  046125  020104                     .ASCIZ  /MULD (R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
066030                                  EM265:
066030  047515  042104  024040                 .ASCIZ  /MODD (R0),AC0 FAILED IN THE INTERRUPT CORRECT FLOWS.
```

```
066115     103  051117  042522  EM266:  .ASCIZ  'CORRECT FLOWS INTERRUPT TEST MODULE CAUSED UNEXPECTED INTERRUPT.'

                                       ;THESE ARE ERROR DATA TABLE HEADERS:

066216  020040  042524  052123  DH1:    .ASCII  '  TEST.'<TAB>'CALL AT PC.'<TAB>'ERROR AT PC.'<TAB>
066257     107  052117  043040          .ASCIZ  'GOT FPS.'<TAB>'EXPECTED FPS.'
066306  020040  042524  052123  DH2:    .ASCIZ  '  TEST.'<TAB>'CALL AT PC.'<TAB>'ERROR AT PC.'
        066216                  DH3=DH1
        066216                  DH4=DH1
        066216                  DH5=DH1
        066216                  DH6=DH1
        066216                  DH7=DH1
        066216                  DH10=DH1
        066216                  DH11=DH1
        066216                  DH12=DH1
        066216                  DH13=DH1
        066216                  DH14=DH1
        066216                  DH15=DH1
        066216                  DH16=DH1
066347     040  052040  051505  DH17:   .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'<TAB>'FEC.'<TAB>'FPS.'
        066216                  DH20=DH1
066422  020040  042524  052123  DH21:   .ASCII  '  TEST.'<TAB>'PC OF CALL'<TAB>'PC OF ERROR.'<TAB>
066462  047507  020124  042506          .ASCIZ  'GOT FEC.'<TAB>'EXPECTED FEC.'
066511     040  052040  051505  DH22:   .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'<TAB>'FPS.'
        066216                  DH23=DH1
        066216                  DH32=DH1
        066216                  DH24=DH1
        066216                  DH25=DH1
        066216                  DH26=DH1
        066216                  DH27=DH1
        066216                  DH30=DH1
        066216                  DH31=DH1
        066216                  DH33=DH1
        066216                  DH34=DH1
        066216                  DH35=DH1
        066216                  DH36=DH1
        066216                  DH37=DH1
        066216                  DH40=DH1
        066216                  DH41=DH1
        066216                  DH42=DH1
        066216                  DH43=DH1
        066216                  DH44=DH1
        066216                  DH45=DH1
        066216                  DH246=DH1
        066216                  DH46=DH1
        066216                  DH47=DH1
        066216                  DH50=DH1
        066216                  DH51=DH1
        066216                  DH52=DH1
        066216                  DH111=DH1
        066216                  DH112=DH1
        066216                  DH113=DH1
        066216                  DH114=DH1
066557     040  052040  051505  DH115:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'<TAB>
066620  047507  020124  042506          .ASCIZ  'GOT FEC.''GOT FPS.''EXPECTED FPS.'
        066557                  DH116=DH115
```

```
066557                  DH117=DH115
066557                  DH120=DH115
066557                  DH121=DH115
066557                  DH122=DH115
066216                  DH123=DH1
066216                  DH124=DH1
066216                  DH125=DH1
066216                  DH126=DH1
066557                  DH127=DH115
066557                  DH130=DH115
066557                  DH131=DH115
066216                  DH132=DH1
066557                  DH133=DH115
066557                  DH134=DH115
066216                  DH135=DH1
066557                  DH136=DH115
066557                  DH137=DH115
066557                  DH140=DH115
066557                  DH141=DH115
066557                  DH142=DH115
066557                  DH143=DH115
066557                  DH144=DH115
066216                  DH145=DH1
066216                  DH146=DH1
066216                  DH147=DH1
066216                  DH150=DH1
066557                  DH151=DH115
066557                  DH152=DH115
066557                  DH153=DH115
066557                  DH154=DH115
066557                  DH155=DH115
066557                  DH156=DH115
066216                  DH157=DH1
066557                  DH160=DH115
066216                  DH161=DH1
066216                  DH162=DH1
066557                  DH163=DH115
066216                  DH164=DH1
066216                  DH53=DH1
066216                  DH54=DH1
066216                  DH55=DH1
066216                  DH56=DH1
066216                  DH57=DH1
066216                  DH60=DH1
066216                  DH61=DH1
066216                  DH62=DH1
066216                  DH63=DH1
066216                  DH64=DH1
066216                  DH65=DH1
066216                  DH66=DH1
066216                  DH67=DH1
066216                  DH70=DH1
066216                  DH71=DH1
066216                  DH72=DH1
066216                  DH73=DH1
066216                  DH74=DH1
```

```
                066216                  DH75=DH1
                066216                  DH76=DH1
                066216                  DH77=DH1
                066216                  DH100=DH1
                066216                  DH101=DH1
                066216                  DH102=DH1
                066216                  DH103=DH1
                066216                  DH104=DH1
                066216                  DH105=DH1
                066216                  DH106=DH1
                066216                  DH107=DH1
                066216                  DH110=DH1
                066557                  DH165=DH115
                066557                  DH166=DH115
                066557                  DH167=DH115
                066557                  DH170=DH115
                066557                  DH171=DH115
                066557                  DH172=DH115
                066557                  DH173=DH115
                066557                  DH174=DH115
                066557                  DH175=DH115
                066557                  DH176=DH115
                066557                  DH177=DH115
                066557                  DH200=DH115
                066557                  DH201=DH115
                066557                  DH202=DH115
                066557                  DH203=DH115
                066557                  DH204=DH115
                066557                  DH205=DH115
                066557                  DH206=DH115
066656  020040  042524  052123  DH220:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
066716  020040  042524  052123  DH207:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR '
066756  043411  052117  043040          .ASCIZ  <TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
067006  020040  042524  052123  DH210:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
                067006                  DH211=DH210
                066716                  DH212=DH207
                066716                  DH213=DH207
                066716                  DH214=DH207
                066716                  DH215=DH207
                067006                  DH216=DH210
067047     040  052040  051505  DH217:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
067107     011  050106  027123          .ASCIZ  <TAB>'FPS.'<TAB>'FEC.'
                067006                  DH221=DH210
                066656                  DH222=DH220
                067006                  DH223=DH210
                066656                  DH224=DH220
                067006                  DH225=DH210
067122  020040  042524  052123  DH226:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
067161     011  047507  020124          .ASCIZ  <TAB>'GOT FEC.'<TAB>'EXPECTED FEC.'
067211     040  052040  051505  DH227:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
067250  043411  052117  043040          .ASCIZ  <TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
                067122                  DH230=DH226
                067211                  DH231=DH227
                067211                  DH232=DH227
```

```
067300  020040  042524  052123  DH233:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'

067341     040  052040  051505  DH234:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
067401     011  047507  020124          .ASCIZ  <TAB>'GOT R0.'<TAB>'EXPECTED R0.'

067427     040  052040  051505  DH235:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>
067453     120  020103  043117          .ASCII  'PC OF TRAP.'<TAB>

067467     040  052040  051505  DH236:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>
067513     120  020103  043117          .ASCII  'PC OF ERROR.'<TAB>'GOT PC.'
067537     011  054105  042520          .ASCIZ  <TAB>'EXPECTED PC.'

        067341                  DH237=DH234

067555     040  052040  051505  DH240:  .ASCII  '  TEST.'<TAB>'PC OF CALL.'<TAB>
067601     120  020103  043117          .ASCII  'PC OF ERROR.'<TAB>
067616  047507  020124  050106          .ASCIZ  'GOT FPS.'<TAB>'EXPECTED FPS.'

        067300                  DH241=DH233
        067555                  DH242=DH240
        067300                  DH243=DH233
        067300                  DH244=DH233
        067300                  DH245=DH233
067645     040  052040  051505  DH247:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'<TAB>'FEC.'
067712  020040  042524  052123  DH250:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
        067712                  DH251=DH250
        066306                  DH252=DH2
        066216                  DH253=DH1
        066216                  DH254=DH1
        066216                  DH255=DH1
        066216                  DH256=DH1
        066216                  DH257=DH1
        066216                  DH260=DH1
        066216                  DH261=DH1
        066216                  DH262=DH1
        066216                  DH263=DH1
        066216                  DH264=DH1
        066216                  DH265=DH1
067752  020040  042524  052123  DH266:  .ASCIZ  '  TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'

                                ;THESE ARE THE DATA FORMAT SPECIFIERS FOR THE DATA TABLE:
070012     004     000     005  DF1:    .BYTE   4,0,5,0,5,0,5,0,5,5,3,5,5,3
070030     004     000     005  DF2:    .BYTE   4,0,5,0,5,5,3,5,5,3
        070012                  DF3=DF1
        070012                  DF4=DF1
        070012                  DF5=DF1
        070012                  DF6=DF1
        070012                  DF7=DF1
        070012                  DF10=DF1
        070012                  DF11=DF1
        070012                  DF12=DF1
        070012                  DF13=DF1
        070012                  DF14=DF1
        070012                  DF15=DF1
```

```
                070012                  DF16=DF1
        070042      004     000     005 DF17:   .BYTE   4,0,5,0,5,0,0
        070051      004     000     005 DF20:   .BYTE   4,0,5,0,5,0,5,0
                070051                  DF21=DF20
                070051                  DF22=DF20
        070061      004     000     005 DF23:   .BYTE   4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2,5,5,2
                070061                  DF24=DF23
                070061                  DF32=DF23
                070061                  DF25=DF23
                070061                  DF26=DF23
                070061                  DF27=DF23
                070061                  DF30=DF23
                070061                  DF31=DF23
        070105      004     000     005 DF33:   .BYTE   4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3,5,5,3
                070105                  DF34=DF33
                070105                  DF35=DF33
                070105                  DF36=DF33
                070061                  DF37=DF23
                070061                  DF40=DF23
                070061                  DF41=DF23
                070061                  DF42=DF23
                070061                  DF43=DF23
                070061                  DF44=DF23
                070061                  DF45=DF23
                070105                  DF246=DF33
                070105                  DF46=DF33
                070105                  DF47=DF33
                070105                  DF50=DF33
                070105                  DF51=DF33
                070105                  DF52=DF33
                070061                  DF111=DF23
                070061                  DF112=DF23
                070061                  DF113=DF23
                070061                  DF114=DF23
        070131      004     000     005 DF115:  .BYTE   4,0,5,0,5,0,0,0,5,5,2,5,5,2,5,5,2,5,5,2
                070131                  DF116=DF115
                070131                  DF117=DF115
                070131                  DF120=DF115
                070131                  DF121=DF115
                070131                  DF122=DF115
                070105                  DF123=DF33
                070105                  DF124=DF33
                070105                  DF125=DF33
                070105                  DF126=DF33
        070155      004     000     005 DF127:  .BYTE   4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3,5,5,3
                070155                  DF130=DF127
                070155                  DF131=DF127
                070105                  DF132=DF33
                070155                  DF133=DF127
                070155                  DF134=DF127
                070105                  DF135=DF33
                070155                  DF136=DF127
                070131                  DF137=DF115
                070131                  DF140=DF115
                070131                  DF141=DF115
                070131                  DF142=DF115
```

```
                070131                  DF143=DF115
                070131                  DF144=DF115
                070061                  DF145=DF23
                070061                  DF146=DF23
                070061                  DF147=DF23
                070061                  DF150=DF23
                070155                  DF151=DF127
                070155                  DF152=DF127
                070155                  DF153=DF127
                070155                  DF154=DF127
                070155                  DF155=DF127
                070155                  DF156=DF127
                070105                  DF157=DF33
                070155                  DF160=DF127
                070105                  DF161=DF33
                070105                  DF162=DF33
                070155                  DF163=DF127
                070105                  DF164=DF33
070201     004      000      005  DF53:    .BYTE   4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2
                070201                  DF54=DF53
                070201                  DF55=DF53
                070201                  DF56=DF53
                070201                  DF57=DF53
                070201                  DF60=DF53
                070201                  DF61=DF53
                070201                  DF62=DF53
                070201                  DF63=DF53
                070201                  DF64=DF53
                070201                  DF65=DF53
                070201                  DF66=DF53
                070201                  DF67=DF53
070233     004      000      005  DF70:    .BYTE   4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3
                070233                  DF71=DF70
                070233                  DF72=DF70
                070233                  DF73=DF70
                070233                  DF74=DF70
                070233                  DF75=DF70
                070233                  DF76=DF70
                070233                  DF77=DF70
                070233                  DF100=DF70
                070233                  DF101=DF70
                070233                  DF102=DF70
                070233                  DF103=DF70
                070233                  DF104=DF70
                070233                  DF105=DF70
                070233                  DF106=DF70
                070233                  DF107=DF70
                070233                  DF110=DF70
070265     004      000      005  DF165:   .BYTE   4,0,5,0,5,0,0,0,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2
                070265                  DF166=DF165
                070265                  DF167=DF165
                070265                  DF170=DF165
                070265                  DF171=DF165
                070265                  DF172=DF165
                070265                  DF173=DF165
                070265                  DF174=DF165
```

```
                070265                  DF175=DF165
070317          004     000     005     DF176:  .BYTE   4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3
                070317                  DF177=DF176
                070317                  DF200=DF176
                070317                  DF201=DF176
                070317                  DF202=DF176
                070317                  DF203=DF176
                070317                  DF204=DF176
                070317                  DF205=DF176
                070317                  DF206=DF176
070351          004     000     005     DF210:  .BYTE   4,0,5,0,5,0,5,0
070361          004     000     005     DF207:  .BYTE   4,0,5,0,5,5,5,3,5,5,5,3,5,5,5,3,5,5,5,3
                070361                  DF211=DF207
070405          004     000     005     DF212:  .BYTE   4,0,5,0,5,0,5,0,5,5,5,3,5,5,5,3,5,5,5,3,5,5,5,3
                070405                  DF213=DF212
                070405                  DF214=DF212
                070405                  DF215=DF212
                070361                  DF216=DF207
070435          004     000     005     DF217:  .BYTE   4,0,5,0,5,0,0
                070361                  DF220=DF207
                070361                  DF221=DF207
                070361                  DF222=DF207
                070361                  DF223=DF207
                070361                  DF224=DF207
                070361                  DF225=DF207
                070405                  DF226=DF212
                070405                  DF227=DF212
                070405                  DF230=DF212
                070405                  DF231=DF212
                070405                  DF232=DF212
070444          004     000     005     DF233:  .BYTE   4,0,5,0,5,5,3,5,5,3,5,5,3
070461          004     000     005     DF234:  .BYTE   4,0,5,0,5,0,0

070470          004     000     005     DF235:  .BYTE   4,0,5,0
                070461                  DF236=DF234
                070461                  DF237=DF234
                070461                  DF240=DF234
                070444                  DF241=DF233
                070461                  DF242=DF234
                070444                  DF243=DF233
                070444                  DF244=DF233
                070444                  DF245=DF233
070474          004     000     005     DF247:  .BYTE   4,0,5,0,5,0
                070474                  DF250=DF247
                070474                  DF251=DF247

070502          004     000     005     DF252:  .BYTE   4,0,5,0
070506          004     000     005     DF253:  .BYTE   4,0,5,0,5,0,5,0,5,5,0,5,5,0,5,5,0,5,5,0,5,5,0,5,5,3,5,5,3
                070506                  DF254=DF253
                070506                  DF255=DF253
                070506                  DF256=DF253
                070506                  DF257=DF253
                070506                  DF260=DF253
                070506                  DF261=DF253
                070506                  DF262=DF253
                070506                  DF263=DF253
```

```
                070506                  DF264=DF253
                070506                  DF265=DF253
                070502                  DF266=DF252


                                                .EVEN

                                        ;THESE ARE ERROR DATA TABLES (FORMATTED ABOVE):
070540  001232  001234  037151  DT1:    .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP5,$TAB,$TMP6
070562  001313  037156  001240          .WORD   $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4,0
070576  001232  001234  037151  DT2:    .WORD   $TMP0,$TMP1,$TAB,$TMP2,$CRLF,MS3,$TMP3,$CRLF,MS4,$TMP4,0
                070540                  DT3=DT1
                070540                  DT4=DT1
                070540                  DT5=DT1
                070540                  DT6=DT1
                070540                  DT7=DT1
                070540                  DT10=DT1
                070540                  DT11=DT1
                070540                  DT12=DT1
                070540                  DT13=DT1
                070540                  DT14=DT1
                070540                  DT15=DT1
                070540                  DT16=DT1
070624  001232  001234  037151  DT17:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TMP5,0
070644  001232  001234  037151  DT20:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TAB,$TMP5,0
070666  001232  001234  037151  DT21:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TAB,$TMP3,0
070710  001232  001234  037151  DT22:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP5,0
070726  001232  001234  037151  DT23:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP7,$TAB,$TMP10,$CRLF,MS1,$TMP3,$CRLF,MS2,
070762  001313  037271  001244          .WORD   $CRLF,MS5,$TMP5,$CRLF,MS6,$TMP6,0
                070726                  DT32=DT23
                070726                  DT24=DT23
                070726                  DT25=DT23
                070726                  DT26=DT23
                070726                  DT27=DT23
                070726                  DT30=DT23
                070726                  DT31=DT23
                070726                  DT33=DT23
                070726                  DT34=DT23
                070726                  DT35=DT23
                070726                  DT36=DT23
                070726                  DT37=DT23
                070726                  DT40=DT23
                070726                  DT41=DT23
                070726                  DT42=DT23
                070726                  DT43=DT23
                070726                  DT44=DT23
                070726                  DT45=DT23
                070726                  DT246=DT23
                070726                  DT46=DT23
                070726                  DT47=DT23
                070726                  DT50=DT23
                070726                  DT51=DT23
                070726                  DT52=DT23
                070726                  DT111=DT23
                070726                  DT112=DT23
```

```
                070726                  DT113=DT23
                070726                  DT114=DT23
071000  001232  001234  037151  DT115:  .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP11,$TMP7,$TAB,$TMP13
071022  001313  037156  001240          .WORD   $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4
071056  001313  037271  001244          .WORD   $CRLF,MS5,$TMP5,$CRLF,MS6,$TMP6,0
                071000                  DT116=DT115
                071000                  DT117=DT115
                071000                  DT120=DT115
                071000                  DT121=DT115
                071000                  DT122=DT115
                070726                  DT123=DT23
                070726                  DT124=DT23
                070726                  DT125=DT23
                070726                  DT126=DT23
                071000                  DT127=DT115
                071000                  DT130=DT115
                071000                  DT131=DT115
                070726                  DT132=DT23
                071000                  DT133=DT115
                071000                  DT134=DT115
                070726                  DT135=DT23
                071000                  DT136=DT115
                071000                  DT137=DT115
                071000                  DT140=DT115
                071000                  DT141=DT115
                071000                  DT142=DT115
                071000                  DT143=DT115
                071000                  DT144=DT115
                070726                  DT145=DT23
                070726                  DT146=DT23
                070726                  DT147=DT23
                070726                  DT150=DT23
                071000                  DT151=DT115
                071000                  DT152=DT115
                071000                  DT153=DT115
                071000                  DT154=DT115
                071000                  DT155=DT115
                071000                  DT156=DT115
                070726                  DT157=DT23
                071000                  DT160=DT115
                070726                  DT161=DT23
                070726                  DT162=DT23
                071000                  DT163=DT115
                070726                  DT164=DT23
071054  001232  001234  037151  DT53:   .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP11,$TAB,$TMP13
071074  001313  037156  001240          .WORD   $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4
071110  001313  037377  001244          .WORD   $CRLF,MS11,$TMP5,$CRLF,MS12,$TMP6
071124  001313  037331  001250          .WORD   $CRLF,MS7,$TMP7,$CRLF,MS10,$TMP10,0
                071054                  DT54=DT53
                071054                  DT55=DT53
                071054                  DT56=DT53
                071054                  DT57=DT53
                071054                  DT60=DT53
                071054                  DT61=DT53
                071054                  DT62=DT53
                071054                  DT63=DT53
```

```
                071054                  DT64=DT53
                071054                  DT65=DT53
                071054                  DT66=DT53
                071054                  DT67=DT53
                071054                  DT70=DT53
                071054                  DT71=DT53
                071054                  DT72=DT53
                071054                  DT73=DT53
                071054                  DT74=DT53
                071054                  DT75=DT53
                071054                  DT76=DT53
                071054                  DT77=DT53
                071054                  DT100=DT53
                071054                  DT101=DT53
                071054                  DT102=DT53
                071054                  DT103=DT53
                071054                  DT104=DT53
                071054                  DT105=DT53
                071054                  DT106=DT53
                071054                  DT107=DT53
                071054                  DT110=DT53
071142  001232  001234  037151  DT165:  .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP13,$TMP11,$TMP12
071162  001313  037156  001240          .WORD   $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4
071176  001313  037377  001244          .WORD   $CRLF,MS11,$TMP5,$CRLF,MS12,$TMP6
071212  001313  037331  001250          .WORD   $CRLF,MS7,$TMP7,$CRLF,MS10,$TMP10,0
                071142                  DT166=DT165
                071142                  DT167=DT165
                071142                  DT170=DT165
                071142                  DT171=DT165
                071142                  DT172=DT165
                071142                  DT173=DT165
                071142                  DT174=DT165
                071142                  DT175=DT165
                071142                  DT176=DT165
                071142                  DT177=DT165
                071142                  DT200=DT165
                071142                  DT201=DT165
                071142                  DT202=DT165
                071142                  DT203=DT165
                071142                  DT204=DT165
                071142                  DT205=DT165
                071142                  DT206=DT165

071230  001232  001234  037151  DT210:  .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP3
071244  037151  001242  000000          .WORD   $TAB,$TMP4,0
071252  001232  001234  037151  DT207:  .WORD   $TMP0,$TMP1,$TAB,$TMP2,$CRLF,MS41,$CRLF,$TMP3
071272  001313  037544  001313          .WORD   $CRLF,MS42,$CRLF,$TMP4,$CRLF,MS43,$CRLF,$TMP5
071312  001313  037577  001313          .WORD   $CRLF,MS44,$CRLF,$TMP6,0
                071252                  DT211=DT207
071324  001232  001234  037151  DT212:  .WORD   $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP10,$TAB,$TMP11
071344  001313  037520  001313          .WORD   $CRLF,MS41,$CRLF,$TMP3,$CRLF,MS42,$CRLF,$TMP4
071364  001313  037562  001313          .WORD   $CRLF,MS43,$CRLF,$TMP5,$CRLF,MS44,$CRLF,$TMP6,0
        071324                  DT213=DT212
        071324                  DT214=DT212
        071324                  DT215=DT212
        071252                  DT216=DT207
```

```
071406  001232  001234  037151  DT217:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP3,$TMP4,0
        071252                  DT220=DT207
        071252                  DT221=DT207
        071252                  DT222=DT207
        071252                  DT223=DT207
        071252                  DT224=DT207
        071252                  DT225=DT207
        071324                  DT226=DT212
        071324                  DT227=DT212
        071324                  DT230=DT212
        071230                  DT231=DT210
        071252                  DT232=DT207



071426  001232  001234  037151  DT233:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,$CRLF
071440  037446  001244  001313          .WORD  MS37,$TMP5,$CRLF,MS42,$TMP6
071452  001313  037520  001250          .WORD  $CRLF,MS41,$TMP7,0

071462  001232  001234  037151  DT234:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP3,$TMP4,0

071502  001232  001234  037151  DT235:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,0

        071462                  DT236=DT234
        071462                  DT237=DT234
        071462                  DT240=DT234
        071426                  DT241=DT233
        071462                  DT242=DT234
        071426                  DT243=DT233
        071426                  DT244=DT233
        071426                  DT245=DT233
071514  001232  001234  037151  DT247:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP3,0
071532  001232  001234  037151  DT250:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,0
        071532                  DT251=DT250

071544  001232  001234  037151  DT252:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,0

071556  001232  001234  037151  DT253:  .WORD  $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP7,$TAB,$TMP10
071576  001313  036760  001254          .WORD  $CRLF,MSA1,$TMP11,$CRLF,MSA2,$TMP12
071612  001313  037021  001244          .WORD  $CRLF,MSA3,$TMP5,$CRLF,MSA4,$TMP6
071626  001313  037051  001240          .WORD  $CRLF,MSA5,$TMP3,$CRLF,MSA6,$TMP4,0
        071556                  DT254=DT253
        071556                  DT255=DT253
        071556                  DT256=DT253
        071556                  DT257=DT253
        071556                  DT260=DT253
        071556                  DT261=DT253
        071556                  DT262=DT253
        071556                  DT263=DT253
        071556                  DT264=DT253
        071556                  DT265=DT253

        071544                  DT266=DT252
```

```
                                ;12345
000001                                  .END
```

```
AAADON  014422
AAA1    013420
AAA10   014036
AAA11   014074
AAA12   014132
AAA13   014170
AAA2    013456
AAA3    013514
AAA4    013552
AAA5    013610
AAA6    013646
AAA7    013704
AAA8    013742
AAA9    014000
ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUCP= 000000
AC0   =%000000
AC1   =%000001
AC2   =%000002
AC3   =%000003
AC4   =%000004
AC5   =%000005
AC6   =%000006
AC7   =%000007
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSP0= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
BBBDON  015124
BBBER1  014700
BBBER2  014764
BBBER3  015012
BBBER4  015040
BBBP1   015104
BBBP2   015114
BBB0    014430
BBB1    014464
BBB2    014512
BBB3    014542
BBB4    014570
BBB5    014626
BBB6    014634
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BPTVEC= 000014
CCCDON  016134
CCC1    015130
CCC10   015524
CCC11   015560
CCC12   015614
CCC13   015650
CCC2    015164
CCC3    015220
CCC4    015254
CCC5    015310
CCC6    015344
CCC7    015400
CCC8    015434
CCC9    015470
CKSWR = 104406
CMPSUB  014232
CMPTMP  014412
CNT   = 000267
CORDON  033146
CORFLG  033130
CORINT  033142
CORMES  037621
CORSUB  032646
CORTMP  033132
CORTRP  033144
CORTV   032744
CORTVO  033106
CORTVI  033112
COR1    032014
COR10   032436
COR11   032502
COR12   032542
COR13   032602
COR2    032040
COR3    032054
COR33   032066
COR4    032126
COR5    032166
COR6    032226
COR7    032272
COR8    032332
COR9    032376
CPSPUR  036646
CPTWO   036664
CR    = 000015
CRLF  = 000200
DDDDON  017070
DDC1    016140
DDD2    016214
DDD3    016270
DDD4    016344
DDD5    016420
DDD6    016474
DDD7    016550
DDISP = 177570
DF1     070012
DF10  = 070012
DF100 = 070233
DF101 = 070233
DF102 = 070233
DF103 = 070233
DF104 = 070233
DF105 = 070233
DF106 = 070233
DF107 = 070233
DF11  = 070012
DF110 = 070233
DF111 = 070061
DF112 = 070061
DF113 = 070061
DF114 = 070061
DF115   070131
DF116 = 070131
DF117 = 070131
DF12  = 070012
DF120 = 070131
DF121 = 070131
DF122 = 070131
DF123 = 070105
DF124 = 070105
DF125 = 070105
DF126 = 070105
DF127   070155
DF13  = 070012
DF130 = 070155
DF131 = 070155
DF132 = 070105
DF133 = 070155
DF134 = 070155
DF135 = 070105
DF136 = 070155
DF137 = 070131
DF14  = 070012
DF140 = 070131
DF141 = 070131
DF142 = 070131
DF143 = 070131
DF144 = 070134
DF145 = 070061
DF146 = 070061
DF147 = 070061
DF15    070012
DF150 = 070061
DF151 = 070155
DF152 = 070155
DF153 = 070155
DF154 = 070155
DF155 = 070155
DF156 = 070155
DF157 = 070105
DF16    070012
DF160 = 070155
DF161 = 070105
DF162 = 070105
DF163 = 070155
DF164 = 070105
DF165   070265
DF166 = 070265
DF167 = 070265
DF17    070042
DF170 = 070265
DF171 = 070265
DF172 = 070265
DF173 = 070265
DF174 = 070265
DF175 = 070265
DF176   070317
DF177 = 070317
DF2     070030
DF20    070051
DF200 = 070317
DF201 = 070317
DF202 = 070317
DF203 = 070317
DF204 = 070317
DF205 = 070317
DF206 = 070317
DF207   070361
DF21  = 070051
DF210   070351
DF211 = 070361
DF212   070405
DF213 = 070405
DF214 = 070405
DF215 = 070405
DF216 = 070361
DF217   070435
DF22  = 070051
DF220 = 070361
DF221 = 070361
```

```
DF222 = 070361
DF223 = 070361
DF224 = 070361
DF225 = 070361
DF226 = 070405
DF227 = 070405
DF23    070061
DF230 = 070405
DF231 = 070405
DF232 = 070405
DF233   070444
DF234   070461
DF235   070470
DF236 = 070461
DF237 = 070461
DF24  = 070061
DF240 = 070461
DF241 = 070444
DF242 = 070461
DF243 = 070444
DF244 = 070444
DF245 = 070444
DF246 = 070105
DF247   070474
DF25  = 070061
DF250 = 070474
DF251 = 070474
DF252   070502
DF253   070506
DF254 = 070506
DF255 = 070506
DF256 = 070506
DF257 = 070506
DF26  = 070061
DF260 = 070506
DF261 = 070506
DF262 = 070506
DF263 = 070506
DF264 = 070506
DF265 = 070506
DF266 = 070502
DF27  = 070061
DF3   = 070012
DF30  = 070061
DF31  = 070061
DF32  = 070061
DF33    070105
DF34  = 070105
DF35  = 070105
DF36  = 070105
DF37  = 070061
DF4   = 070012
DF40  = 070061
DF41  = 070061
DF42  = 070061
DF43  = 070061
DF44  = 070061
DF45  = 070061
DF46  = 070105
DF47  = 070105
DF5   = 070012
DF50  = 070105
DF51  = 070105
DF52  = 070105
DF53    070201
DF54  = 070201
DF55  = 070201
DF56  = 070201
DF57  = 070201
DF6   = 070012
DF60  = 070201
DF61  = 070201
DF62  = 070201
DF63  = 070201
DF64  = 070201
DF65  = 070201
DF66  = 070201
DF67  = 070201
DF7   = 070012
DF70    070233
DF71  = 070233
DF72  = 070233
DF73  = 070233
DF74  = 070233
DF75  = 070233
DF76  = 070233
DF77  = 070233
DH1     066216
DH10  = 066216
DH100 = 066216
DH101 = 066216
DH102 = 066216
DH103 = 066216
DH104 = 066216
DH105 = 066216
DH106 = 066216
DH107 = 066216
DH11  = 066216
DH110 = 066216
DH111 = 066216
DH112 = 066216
DH113 = 066216
DH114 = 066216
DH115   066557
DH116 = 066557
DH117 = 066557
DH12  = 066216
DH120 = 066557
DH121 = 066557
DH122 = 066557
DH123 = 066216
DH124 = 066216
DH125 = 066216
DH126 = 066216
DH127 = 066557
DH13  = 066216
DH130 = 066557
DH131 = 066557
DH132 = 066216
DH133 = 066557
DH134 = 066557
DH135 = 066216
DH136 = 066557
DH137 = 066557
DH14  = 066216
DH140 = 066557
DH141 = 066557
DH142 = 066557
DH143 = 066557
DH144 = 066557
DH145 = 066216
DH146 = 066216
DH147 = 066216
DH15  = 066216
DH150 = 066216
DH151 = 066557
DH152 = 066557
DH153 = 066557
DH154 = 066557
DH155 = 066557
DH156 = 066557
DH157 = 066216
DH16  = 066216
DH160 = 066557
DH161 = 066216
DH162 = 066216
DH163 = 066557
DH164 = 066216
DH165 = 066557
DH166 = 066557
DH167 = 066557
DH17    066347
DH170 = 066557
DH171 = 066557
DH172 = 066557
DH173 = 066557
DH174 = 066557
DH175 = 066557
DH176 = 066557
DH177 = 066557
DH2     066306
DH20  = 066216
DH200 = 066557
DH201 = 066557
DH202 = 066557
DH203 = 066557
DH204 = 066557
DH205 = 066557
DH206 = 066557
DH207   066716
DH21    066422
DH210   067006
DH211 = 067006
DH212 = 066716
DH213 = 066716
DH214 = 066716
DH215 = 066716
DH216 = 067006
DH217   067047
DH22    066511
DH220   066656
DH221 = 067006
DH222 = 066656
DH223 = 067006
DH224 = 066656
DH225 = 067006
DH226   067122
DH227   067211
DH23  = 066216
DH230 = 067122
DH231 = 067211
DH232 = 067211
DH233   067300
DH234   067341
DH235   067427
DH236   067467
DH237 = 067341
DH24  = 066216
DH240   067555
DH241 = 067300
DH242 = 067555
DH243 = 067300
DH244 = 067300
DH245 = 067300
DH246 = 066216
DH247   067645
DH25  = 066216
DH250   067712
DH251 = 067712
DH252 = 066306
DH253 = 066216
DH254 = 066216
DH255 = 066216
DH256 = 066216
DH257 = 066216
DH26  = 066216
DH260 = 066216
DH261 = 066216
DH262 = 066216
DH263 = 066216
DH264 = 066216
DH265 = 066216
DH266   067752
DH27  = 066216
DH3   = 066216
DH30  = 066216
DH31  = 066216
DH32  = 066216
DH33  = 066216
DH34  = 066216
DH35  = 066216
DH36  = 066216
DH37  = 066216
DH4   = 066216
DH40  = 066216
DH41  = 066216
DH42  = 066216
DH43  = 066216
DH44  = 066216
DH45  = 066216
DH46  = 066216
DH47  = 066216
DH5   = 066216
DH50  = 066216
DH51  = 066216
DH52  = 066216
DH53  = 066216
DH54  = 066216
DH55  = 066216
DH56  = 066216
DH57  = 066216
DH6   = 066216
DH60  = 066216
DH61  = 066216
DH62  = 066216
DH63  = 066216
DH64  = 066216
DH65  = 066216
DH66  = 066216
DH67  = 066216
DH7   = 066216
DH70  = 066216
DH71  = 066216
DH72  = 066216
DH73  = 066216
```

```
DH74  = 066216
DH75  = 066216
DH76  = 066216
DH77  = 066216
DISPLA  001142
DISPRE  000174
DIVDSU  016630
DIVDT   017060
DIVFSU  015710
DIVFT   016124
DSWR  = 177570
DT1     070540
DT10  = 070540
DT100 = 071054
DT101 = 071054
DT102 = 071054
DT103 = 071054
DT104 = 071054
DT105 = 071054
DT106 = 071054
DT107 = 071054
DT11  = 070540
DT110 = 071054
DT111 = 070726
DT112 = 070726
DT113 = 070726
DT114 = 070726
DT115   071000
DT116 = 071000
DT117 = 071000
DT12  = 070540
DT120 = 071000
DT121 = 071000
DT122 = 071000
DT123 = 070726
DT124 = 070726
DT125 = 070726
DT126 = 070726
DT127 = 071000
DT13  = 070540
DT130 = 071000
DT131 = 071000
DT132 = 070726
DT133 = 071000
DT134 = 071000
DT135 = 070726
DT136 = 071000
DT137 = 071000
DT14  = 070540
DT140 = 071000
DT141 = 071000
DT142 = 071000
DT143 = 071000
DT144 = 071000
DT145 = 070726
DT146 = 070726
DT147 = 070726
DT15  = 070540
DT150 = 070726
DT151 = 071000
DT152 = 071000
DT153 = 071000
DT154 = 071000
DT155 = 071000
DT156 = 071000
DT157 = 070726
DT16  = 070540
DT160 = 071000
DT161 = 070726
DT162 = 070726
DT163 = 071000
DT164 = 070726
DT165   071142
DT166 = 071142
DT167 = 071142
DT17    070624
DT170 = 071142
DT171 = 071142
DT172 = 071142
DT173 = 071142
DT174 = 071142
DT175 = 071142
DT176 = 071142
DT177 = 071142
DT2     070576
DT20    070644
DT200 = 071142
DT201 = 071142
DT202 = 071142
DT203 = 071142
DT204 = 071142
DT205 = 071142
DT206 = 071142
DT207   071252
DT21    070666
DT210   071230
DT211 = 071252
DT212   071324
DT213 = 071324
DT214 = 071324
DT215 = 071324
DT216 = 071252
DT217   071406
DT22    070710
DT220 = 071252
DT221 = 071252
DT222 = 071252
DT223 = 071252
DT224 = 071252
DT225 = 071252
DT226 = 071324
DT227 = 071324
DT23    070726
DT230 = 071324
DT231 = 071230
DT232 = 071252
DT233   071426
DT234   071462
DT235   071502
DT236 = 071462
DT237 = 071462
DT24  = 070726
DT240 = 071462
DT241 = 071426
DT242 = 071462
DT243 = 071426
DT244 = 071426
DT245 = 071426
DT246 = 070726
DT247   071514
DT25  = 070726
DT250   071532
DT251 = 071532
DT252   071544
DT253   071556
DT254 = 071556
DT255 = 071556
DT256 = 071556
DT257 = 071556
DT26  = 070726
DT260 = 071556
DT261 = 071556
DT262 = 071556
DT263 = 071556
DT264 = 071556
DT265 = 071556
DT266 = 071544
DT27  = 070726
DT3   = 070540
DT30  = 070726
DT31  = 070726
DT32  = 070726
DT33  = 070726
DT34  = 070726
DT35  = 070726
DT36  = 070726
DT37  = 070726
DT4   = 070540
DT40  = 070726
DT41  = 070726
DT42  = 070726
DT43  = 070726
DT44  = 070726
DT45  = 070726
DT46  = 070726
DT47  = 070726
DT5   = 070540
DT50  = 070726
DT51  = 070726
DT52  = 070726
DT53    071054
DT54  = 071054
DT55  = 071054
DT56  = 071054
DT57  = 071054
DT6   = 070540
DT60  = 071054
DT61  = 071054
DT62  = 071054
DT63  = 071054
DT64  = 071054
DT65  = 071054
DT66  = 071054
DT67  = 071054
DT7   = 070540
DT70  = 071054
DT71  = 071054
DT72  = 071054
DT73  = 071054
DT74  = 071054
DT75  = 071054
DT76  = 071054
DT77  = 071054
EEEDON  020100
EEE1    017074
EEE10   017470
EEE11   017524
EEE12   017560
EEE13   017614
EEE2    017130
EEE3    017164
EEE4    017220
EEE5    017254
EEE6    017310
EEE7    017344
EEE8    017400
EEE9    017434
EMTVEC= 000030
EM1     037660
EM10    040477
EM100   053635
EM101   053720
EM102   054...
EM103   0543..
EM104   054422
EM105   054567
EM106   054676
EM107   055005
EM11    040604
EM110   055114
EM111   04373:
EM112   044006
EM113   044064
EM114   044142
EM115   044221
EM116   044277
EM117   044356
EM12    040713
EM120   044514
EM121   044652
EM122   045007
EM123   045144
EM124   045222
EM125   045301
EM126   045357
EM127   045436
EM13    041022
EM130   045514
EM131   045573
EM132   045731
EM133   046067
EM134   046225
EM135   046362
EM136   046517
EM137   046654
EM14    041136
EM140   046742
EM141 = 043731
EM142 = 044006
EM143   047031
EM144   047107
EM145   047166
EM146   047333
EM147   047500
EM15    041250
EM150   047644
EM151   050010
EM152   050076
EM153 = 045144
EM154 = 045222
EM155   050165
EM156   050243
EM157   050322
EM16    041362
EM160   050467
```

```
EM161   050625
EM162   050772
EM163   051136
EM164   051273
EM165   055224
EM166   055310
EM167   055373
EM17    041457
EM170   055425
EM171   055513
EM172   055636
EM173   055723
EM174   056071
EM175   056237
EM176   056351
EM177   056435
EM2     037712
EM20    041534
EM200   056520
EM201   056552
EM202   056641
EM203   057003
EM204   057145
EM205   057233
EM206   057401
EM207   057547
EM21    041566
EM210   057611
EM211   057653
EM212   060017
EM213   060203
EM214   060367
EM215   060553
EM216   060737
EM217   061121
EM22    041620
EM220   061163
EM221   061413
EM222   061657
EM223   062110
EM224   062355
EM225   062606
EM226   063053
EM227   063206
EM23    041701
EM230   063341
EM231   063475
EM232 = 057611
EM233   063631
EM234   063670
EM235   063754
EM236   064022
EM237   064055
EM24    041756
EM240   064141
EM241   064175
EM242   064300
EM243   064334
EM244   064437
EM245   064476
EM246   043341
EM247   064560
EM25    042054
EM250   064614
EM251   064646
EM252   064701
EM253   064772
EM254   065056
EM255   065143
EM256   065231
EM257   065320
EM26    042141
EM260   065406
EM261   065475
EM262   065565
EM263   065656
EM264   065743
EM265   066030
EM266   066115
EM27  = 042054
EM3     037746
EM30    042237
EM31    042311
EM32    041724
EM33    042356
EM34    042401
EM35    042433
EM36    042506
EM37    042554
EM4     040053
EM40    042577
EM41    042631
EM42    042704
EM43    043035
EM44    043166
EM45    043234
EM46    043307
EM47    043364
EM5     040160
EM50    043520
EM51    043573
EM52    043641
EM53    051471
EM54    051522
EM55    051437
EM56    051552
EM57    051643
EM6     040265
EM60    051733
EM61    052035
EM62    052231
EM63    052425
EM64    052536
EM65    052655
EM66    052770
EM67    053037
EM7     040372
EM70    053122
EM71    053153
EM72    053203
EM73    053203
EM74    053235
EM75    053344
EM76    053435
EM77    053525
ERM10   004156
ERRVEC= 000004
ERTYPE  036140
ERT1    036324
ERT2    036560
ERT3    036564
ERT4    036576
ERT5    036610
FFFDON  020630
FFF1    020104
FFF2    020160
FFF3    020234
FFF4    020310
FPSPUR  036614
FPVECT= 000244
GGDATO  011554
GGDONE  011674
GGER0   010052
GGER1   010150
GGER10  010662
GGER11  010730
GGER12  010776
GGER13  011044
GGER14  011112
GGER15  011210
GGER16  011256
GGER17  011324
GGER18  011372
GGER19  011440
GGER2   010216
GGER20  011506
GGER3   010264
GGER4   010332
GGER5   010334
GGER6   010402
GGER7   010450
GGER8   010546
GGER9   010614
GGGDON  025712
GGG1    024240
GGG10   025010
GGG11   025060
GGG12   025130
GGG13   025200
GGG14   025250
GGG2    024310
GGG3    024360
GGG4    024430
GGG5    024500
GGG6    024550
GGG7    024620
GGG8    024670
GGG9    024740
GGP1    011564
GGP2    011574
GGP3    011604
GGP4    011614
GGP5    011624
GGP6    011634
GGP7    011644
GGP8    011654
GGP9    011664
GG1     006762
GG10    007256
GG11    007314
GG12    007336
GG13    007346
GG14    007364
GG15    007422
GG16    007440
GG17    007506
GG18    007516
GG19    007552
GG2     007020
GG20    007610
GG21    007632
GG22    007642
GG23    007660
GG24    007716
GG25    007734
GG26    010002
GG27    010012
GG28    010046
GG3     007042
GG4     007052
GG5     007070
GG6     007126
GG7     007144
GG8     007212
GG9     007222
G$SWR = 104405
HHDATO  006606
HHDONE  006756
HHERO   005626
HHERO0  005742
HHER1   005674
HHER10  006470
HHER2   006010
HHER3   006056
HHER4   006124
HHER5   006172
HHER6   006240
HHER7   006306
HHER8   006354
HHER9   006422
HHHDON  030020
HHH1    025716
HHH10   027016
HHH11   027116
HHH12   027216
HHH13   027316
HHH2    026016
HHH3    026116
HHH4    026216
HHH5    026316
HHH6    026416
HHH7    026516
HHH8    026616
HHH9    026716
HHP0    006616
HHP1    006626
HHP10   006736
HHP11   006746
HHP2    006636
HHP3    006646
HHP4    006656
HHP5    006666
HHP6    006676
HHP7    006706
HHP8    006716
HHP9    006726
HHTRAP  006536
HH1     005036
HH10    005250
HH11    005266
HH12    005316
HH13    005340
HH14    005360
HH15    005370
HH16    005376
```

```
##17     005414
##18     005452
##19     005474
##2      005066
##20     005504
##21     005522
##22     005552
##23     005574
##24     005604
##25     005622
##3      005110
##4      005130
##5      005140
##6      005146
##7      005160
##8      005216
##9      005240
4T    =  000011
HXDATO  013304
HXDONE  013414
HXER1   012672
HXER10  013162
HXER11  013214
HXER12  013234
HXER13  013264
HXER2   012732
HXER22  012746
HXER3   012762
HXER33  012776
HXER4   013012
HXER5   012712
HXER6   013046
HXER66  013062
HXER7   013076
HXER8   013030
HXER9   013126
HXP1    013314
HXP2    013324
HXP3    013334
HXP4    013344
HXP5    013354
HXP6    013364
HXP7    013374
HXP8    013404
HX1     011700
HX10    012106
HX11    012130
HX12    012150
HX13    012160
HX14    012166
HX15    012204
HX16    012232
HX165   012236
HX17    012264
HX18    012322
HX19    012342
HX2     011730
HX20    012352
HX21    012354
HX22    012404
HX23    012424
HX24    012444
HX25    012454
HX26    012462
HX27    012464
HX28    012516
HX29    012540
HX3     011744
HX30    012550
HX31    012566
HX32    012616
HX33    012640
HX34    012650
HX35    012666
HX4     011764
HX5     012004
HX6     012014
HX7     012022
HX8     012040
HX9     012072
IIIDON  021470
IIII    020634
III2    020700
III3    020744
III4    021010
IOTVEC= 000020
JJJDON  022430
JJJ1    021474
JJJ2    021560
JJJ3    021644
JJJ4    021730
KKKDON  023272
KKK1    022434
KKK3    022500
KKK4    022544
KKK5    022610
LF    = 000012
LLLDON  024234
LLL1    023276
LLL2    023362
LLL3    023446
LLL4    023532
LOOP    005034
LPERR = 104413
MMMDON  030746
MMM1    030024
MMM2    030076
MMM3    030150
MMM4    030222
MMM5    030274
MNUMBE= 000266
MODDDO  031754
MODDD1  031764
MODDOV  031366
MODDSU  027422
MODDTO  030000
MODDT1  030010
MODFDO  030726
MODFD1  030736
MODFOV  030352
MODFSU  025324
MODFTO  025662
MODFT1  025672
MODP1   025702
MSA1    036760
MSA2    036776
MSA3    037021
MSA4    037033
MSA5    037051
MSA6    037064
MS1     037156
MS10    037355
MS11    037377
MS12    037423
MS2     037174
MS3     037214
MS37    037446
MS4     037243
MS40    037464
MS41    037520
MS415   037500
MS42    037544
MS43    037562
MS44    037577
MS5     037271
MS6     037313
MS7     037331
MULDSU  020370
MULDT   020620
MULFSU  017654
MULFT   020070
NNNDON  031774
NNN1    030752
NNN2    031054
NNN3    031156
NNN4    031260
OVDNTT  022420
OVDTT   024224
OVFNTT  021460
OVFTT   023262
OVUNDN  022020
OVUNDT  023622
OVUNFN  021060
OVUNFT  022660
PIRQ  = 177772
PIRQVE= 000240
POWERM  037104
PRUGNU= 000002
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PWRVEC= 000024
RDCHR = 104407
RESREG= 104411
RESVEC= 000010
RSETUP= 104412
R6    =%000006
R7    =%000007
SAVREG= 104410
SPACE   037153
STACK = 001100
START   004336
STKLMT= 177774
SWR     001140
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
TAB   = 000011
TBITVE= 000014
TKVEC = 000060
TPVEC = 000064
TRAPVE= 000034
TRTVEC= 000014
TST1    005034
TST10   017072
TST11   020102
TST12   020632
TST13   021472
TST14   022432
TST15   023274
TST16   024236
TST17   025714
TST2    006760
TST20   030022
TST21   030750
TST22   031776
TST23   033150
TST3    011676
TST4    013416
TST5    014424
TST6    015126
TST7    016136
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
$APTHD  004322
$ATYC   035010
$ATY1   034764
$ATY3   034772
$ATY4   035002
$AUTOB  001134
$BASE   001372
$BDADR  001122
$BDDAT  001126
$BELL   001306
$CDW1   001376
$CDW2   001400
$CHARC  034532
$CKSWR  035232
$CLR.T  033352
$CMTAG  001100
$CM1  = 000024
$CM2  = 000050
```

```
$CR3    = 000024        $ERTTL  001112  $MSGTY  001316  $REG3   001170  $TMP20  001272
$CRLF   = 000024        $ESCAP  001304  $MSWR   035646  $REG4   001172  $TMP21  001274
$CNTLO  035641          $ETABL  001336  $MTYP1  001347  $REG5   001174  $TMP22  001276
$CNTLQ  035634          $ETEND  001442  $MTYP2  001353  $REG6   001176  $TMP23  001300
$CPUOP  001344          $FATAL  001320  $MTYP3  001357  $REG7   001200  $TMP3   001240
$CRLF   001313          $FFLG   035230  $MTYP4  001363  $RESRE  034216  $TMP4   001242
$DDMC   001402          $FILLC  001156  $MXCNT  033720  $RTNAD  033432  $TMP5   001244
$DDMI   001424          $FILLS  001155  $NULL   001154  $RTRN   033426  $TMP6   001246
$DDMIC  001426          $GDADR  001120  $NWTST= 000001  $SAVRE  034160  $TMP7   001250
$DDMII  001430          $GDDAT  001124  $OCNT   034760  $SAVR6  036136  $TN     = 000023
$DDMI2  001432          $GET42  033334  $OMODE  034762  $SCOPE  033442  $TPB    001152
$DDMI3  001434          $GTSWR  035302  $OVER   033704  $SETUP= 000137  $TPFLG  001157
$DDMI4  001436          $HC     = 000003  $PASS   001324  $STUP  = 177777  $TPS    001150
$DDMI5  001440          $HIBTS  004322  $PASTM  004330  $SVLAD  033650  $TRAP   035670
$DDMI2  001406          $ICNT   001104  $PWRAD  036114  $SVPC   = 004322  $TRAP2  035712
$DDMI3  001410          $ILLUP  036132  $PWRDN  035754  $SWR    = 177400  $TPP    = 000014
$DDMI4  001412          $INTAG  001135  $PWRMG  036110  $SWREG  001340  $TRPAD  035724
$DDMI5  001414          $ITEMB  001114  $PWRUP  036026  $SWRMK= 000000  $TSTM   004326
$DDMI6  001416          $LF     001314  $QUES   001312  $SWRMS= 000200  $TSTNM  001102
$DDMI7  001420          $LFLG   035227  $RDCHR  035514  $TAB    037151  $TYPE   034254
$DDMI8  001422          $LOOP   033430  $RDSZ   = 000001  $TBIT   033434  $TYPEC  034466
$DDMI9  001424          $LPADR  001106  $REGAD  001160  $TERM   = 000030  $TYPEX  034534
$DEVCT  001326          $LPERR  001110  $REG0   001162  $TESTN  001322  $TYPOC  034562
$DEVTM  001374          $MADR1  001350  $REG1   001164  $TIMES  001302  $TYPON  034576
$ZORGN  033372          $MADR2  001354  $REG10  001202  $TKB    001146  $TYPOS  034536
$ENDADC 033362          $MADR3  001360  $REG11  001204  $TKS    001144  $UNIT   001330
$ENDOCT 033204          $MADR4  001364  $REG12  001206  $TMP0   001232  $UNITM  004332
$ENULL  033436          $MAIL   001316  $REG13  001210  $TMP1   001234  $USWR   001342
$ENV    001336          $MAMS1  001346  $REG14  001212  $TMP10  001252  $VECT1  001366
$ENVM   001337          $MAMS2  001352  $REG15  001214  $TMP11  001254  $VECT2  001370
$EOP    033150          $MAMS3  001356  $REG16  001216  $TMP12  001256  $XTSTR  033454
$EOPCT  033176          $MAMS4  001362  $REG17  001220  $TMP13  001260  $$GET4= 000001
$ERFLG  001103          $MBADR  004324  $REG2   001166  $TMP14  001262  $OFILL  034761
$ERMAX  001115          $MFLG   035226  $REG20  001222  $TMP15  001264  .       = 071644
$ERROR  033722          $MNEW   035657  $REG21  001224  $TMP16  001266  .LPER   036702
$ERRPC  001116          $MSGAD  001332  $REG22  001226  $TMP17  001270  .RSET   036710
$ERPTB  001442          $MSGLG  001334  $REG23  001230  $TMP2   001236  .$X     = 004322

. ABS.  071644      000

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

DSKZ:DFFPBA.BIN,DSKZ:DFFPBA.SEQ/SOL=DFFPBA.P11
RUN-TIME:  85 71 5 SECONDS
RUN-TIME RATIO: 215/162=1.3
CORE USED:  31K  (61 PAGES)
```