I D E N T I F I C A T I O N

PRODUCT CODE:	MAINDEC-11-DQFPD-B-D
PRODUCT NAME:	PDP-11/6X - FP11-F FLOATING POINT UNIT ADD/SUB/MUL/DIV RANDOM OPERAND EXERCISER
DATE :	MAY, 1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	KEN CHAPMAN
REVISED BY:	DON NORTH

CONTENTS

1. ABSTRACT

THIS PROGRAM IS AN EXERCISER FOR THE PDP-11/6X FLOATING POINT ADD, SUBTRACT, MULTIPLY, AND DIVIDE INSTRUCTIONS. RANDOM NUMBER PATTERNS ARE USED AS THE OPERANDS, AND THE HARDWARE GENERATED RESULTS ARE CHECKED AGAINST RESULTS OBTAINED FROM FLOATING POINT SOFTWARE ROUTINES TO INSURE CORRECTNESS. THE PDP-11/6X IS OPERATED IN DOUBLE AND SINGLE FLOATING MODE, ROUND AND TRUNCATE MODE, AND WITH UNDERFLOW AND OVERFLOW CONDITIONS ENABLED AND DISABLED. THE PROGRAM WILL RUN FOR 400(8) "SUBPASSES" BEFORE GIVING AN "END OF PASS" INDICATION, SO THAT A SUFFICIENT NUMBER OF RANDOM PATTERNS ARE OBTAINED FOR USE AS OPERANDS. ALSO AT THIS TIME, OPTIONAL STATUS INFORMATION ON THE TYPES OF RANDOM OPERANDS SELECTED CAN BE PRINTED ON THE CONSOLE. BOTH "HOT" (FP11-E OPTION) AND "WARM" (PDP-11/6X MICROCODE) FLOATING POINT UNITS CAN BE SELECTED FOR TESTING.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11/6X STANDARD COMPUTER WITH MINIMUM 16K OF MEMORY. OPTIONAL FP11-E FLOATING POINT UNIT, IF SELECTED.

2.2 STORAGE

THE PROGRAM USES MEMORY 0-34120(8). THE UPPER 2.0K WORDS ARE RESERVED FOR THE XXDP MONITOR, IF EMPLOYED.

2.3 PRELIMINARY PROGRAMS

THE CPU, CACHE, AND MEMORY TEST PROGRAMS MUST BE RUN FIRST TO VERIFY THE CORRECT OPERATION OF THE BASE MACHINE.

THE PDP-11/6X - FP11-E FLOATING POINT PROCESSOR INSTRUCTION SET TESTS SHOULD THEN BE RUN IN THE FOLLOWING ORDER:

(1) DQFPA FPU BASIC INSTRUCTION TESTS
(2) DQFPB FPU ADVANCED INSTRUCTION TESTS
(3) DQFPC FPU INSTRUCTION EXERCISER
(4) DQFPD FPU ADD/SUB/MUL/DIV RANDOM EXERCISER

3. LOADING PROCEDURE

USE THE STANDARD PROCEDURE FOR ABSOLUTE TAPES, OR LOAD VIA XXDP MEDIA.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1
SWITCH REGISTER (000000) IS WORST CASE TEST.

4.2 STARTING ADDRESS

THE PROGRAM MUST ALWAYS BE STARTED AT LOCATION 200(8).

4.3 PROGRAM/OPERATOR ACTION

LOADING VIA ABSOLUTE PAPERTAPE:

(1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
(2) LOAD ADDRESS 200 (8).
(3) SET SWITCHES (SEE SECTION 5.1)
SR=(000000) IS WORST CASE TEST.
(4) PRESS CONTROL/START TO BEGIN.
(5) PROGRAM TYPES IDENTIFICATION HEADER (VERIFY THAT THE CORRECT PROGRAM HAS BEEN LOADED!), AND EXECUTION BEGINS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE DEFINITION OF THE SPECIFIC BITS IN THE SWITCH REGISTER (EITHER HARDWARE OR SOFTWARE) ARE AS FOLLOWS:

SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRENTLY EXECUTING TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS (WHICH IS AN "ERROR MESSAGE" RESULTING FROM AN ERROR DETECTED IN THE HARDWARE)
SW12=1	010000	INHIBIT STATUS TYPEOUTS (WHICH IS A NON-ERROR RELATED INFORMATIVE MESSAGE, SUCH AS "END PASS #XXX")
SW11=1	004000	INHIBIT ITERATIONS PER TEST
SW10	002000	SET=BELL ON ERROR/CLEAR=BELL ON PASS END
SW09=1	001000	LOOP ON ERROR
SW08=1	000400	LOOP ON TEST NUMBER IN "SLPTST" IF SET, THEN THE TEST SPECIFIED BY THE TEST NUMBER CONTAINED IN THE MEMORY WORD "SLPTST" (SEE PROGRAM LISTING) WILL SPECIFY THE DESIRED TEST ON WHICH TO LOOP.
SW01	000002	CLEAR=TEST HOT-FP/WARM-FP ALTERNATELY EACH PASS (IE, PASS#1 HFP, PASS#1 WFP, PASS#2 HFP, PASS#2 WFP, ETC) SET=TEST ONLY UNIT SPECIFIED IN SW00
SW00	000001	SET=SELECT WARM FP, IF SW01=1

CLEAR=SELECT HOT FP, IF SW01=1

NOTE FOR SW01, SW00 - IF NO HOT FP (FP11-E) IS PRESENT, THEN WARM FP (PDP-11/6X MICROCODE) IS AUTOMATICALLY SELECTED.

5.2 PROGRAM/OPERATOR ACTION

ONCE EXECUTION HAS BEGUN, MINIMAL OPERATOR INTERVENTION IS REQUIRED, UNLESS THE PROGRAM DETECTS AN ERROR IN THE HARDWARE.

IF ALL IS WELL, THE PROGRAM TYPES ITS NAME UPON BEGINNING; AND AT THE START OF EACH PASS, THE CURRENT PASS NUMBER (IN OCTAL) IS ECHOED. NOTE THAT SETTING SW<12>=1 WILL INHIBIT THE TYPEOUT OF THE BEGIN AND END PASS MESSAGES.
IF SW<10>=0, THE CONSOLE BELL WILL BE RUNG AT THE END OF EACH PASS. NOTE THAT ONLY SW<10> AFFECTS THE BELL RINGING AT END OF PASS - SW<12> HAS NO EFFECT ON THIS FUNCTION.

IF AN ERROR OCCURS DURING EXECUTION, MANY VARIATIONS IN ACTION ARE POSSIBLE DEPENDING UPON THE SWITCH SETTINGS.
SW<15>=1 WILL CAUSE THE CPU TO HALT AFTER AN ERROR.
SW<13>=1 WILL ALSO INHIBIT ANY ERROR MESSAGE TYPEOUT THAT WOULD OCCUR AT THIS TIME.
SW<10>=1 WILL CAUSE THE CONSOLE BELL TO BE RUNG ONLY WHEN AN ERROR IS DETECTED (AND NOT AT THE END OF A PASS).
SW<9>=1 CAUSES THE PROGRAM TO LOOP ON THE MOST RECENT ERROR, AS LONG AS IT CONTINUES TO OCCUR.

THERE ARE ALSO SEVERAL OTHER GENERAL USE FUNCTIONS DEFINED BY THE SWITCHES:
SW<11>=1 WILL INHIBIT THE ITERATIONS (=2000(10)) PERFORMED OF EACH TEST ON PASSES 2,3,4, ... THRU THE PROGRAM.
SW<14>=1 CAUSES THE PROGRAM TO LOOP INDEFINATELY ON THE CURRENTLY EXECUTING TEST.
SW<8>=1 CAUSES THE PROGRAM TO CONTINUE EXECUTION AS NORMAL, EXCEPT WHEN THE CONTENTS OF MEMORY WORD "SLPTST" MATCHES THE NUMBER OF THE TEST CURRENTLY EXECUTING. AT THIS POINT, THE TEST IS LOOPED ON INDEFINATELY, UNTIL EITHER SW<8>=0 OR "SLPTST" IS CHANGED. NOTE THAT IF "SLPTST" DOES NOT MATCH THE TEST NUMBER OF ANY TEST, THE CONTENTS OF "SLPTST" ARE EFFECTIVELY IGNORED, AND EXECUTION PROCEEDS NORMALLY.

5.3 HOT (FP11-E) / WARM (PDP-11/6X) SELECTION

WHEN THE PROGRAM IS STARTED (AT 200(8)), A MESSAGE IS OPTIONALLY PRINTED INDICATING THE PRESCENCE/ABSCENCE OF AN FP11-E HOT FLOATING POINT UNIT OPTION (BASED UPON WHETHER "WHAMI" BIT<04> IS 1/0 RESPECTIVELY).

IF NO FP11-E HOT FP OPTION IS PRESENT, THE MESSAGE IS TYPED,

AND ANY ATTEMPTS TO SELECT IT FOR TESTING VIA SW01 AND SW00 ARE IGNORED. ONLY WARM FP (PDP-11/6X MICROCODE) FLOATING POINT CAN BE TESTED/SELECTED.

IF THE FP11-E IS PRESENT, TEST SELECTION IS AS FOLLOWS:

WHEN SW01=0, THE HOT AND WARM FLOATING POINT UNITS ARE TESTED ALTERNATELY EACH PASS - IN THE ORDER (1) HOT, THEN (2) WARM. NOTE THAT EACH "PASS" NOW CONSISTS OF TWO SEPARATE SUB-PASSES.

WHEN SW01=1, THEN DEDICATED SELECTION OF A PARTICULAR UNIT IS SPECIFIED IN SW00:

SW00=0 --> TEST HFP FP11-E OPTION ONLY
SW00=1 --> TEST WFP PDP-11/6X MICROCODE ONLY

6. ERRORS

6.1 FORMAT OF MESSAGES

6.1.1 ALL ERROR MESSAGES CONSIST OF THREE LINES OF DATA:

THE FIRST LINE IS A BRIEF MESSAGE WHICH EXPLAINS WHAT ERROR WAS DETECTED (EG, THE RESULT OF THE "ABSF" INSTRUCTION WAS BAD).

THE PREFIX "HOT:" OR "WARM:" IS ALSO ATTACHED TO THE MESSAGE TO INDICATE THE SOURCE OF THE ERROR; THE FP11-E UNIT OR THE PDP-11/6X RESPECTIVELY.

THE SECOND LINE CONSISTS OF DATA HEADERS TO IDENTIFY THE VALUES TYPED OUT ON LINE THREE. THESE HEADERS WILL EITHER BE OF THE FORM "EXPECTED" AND "RECEIVED" DATA, OR WILL BE A MNEMONIC NAME OF A WORD LOCATION IN MEMORY OR REGISTERS.

THE THIRD LINE DISPLAYS THE CONTENTS OF THE LOCATIONS SPECIFIED BY LINE TWO AS SIX DIGIT OCTAL NUMBERS. NOTE THAT ALL DATA DISPLAYED IN ANY MESSAGES ARE OCTAL NUMBERS.

AS EXPLAINED IN SECTION 5.2, SETTING SW<13>=1 WILL SUPPRESS THE TYPING OF THESE MESSAGES.

6.1.2 FLOATING POINT UNIT DATA FORMATS:

FLOATING POINT STATUS WORD (FPS):

BIT##	OCTAL	FUNCTION
15	100000	FER - FLOATING ERROR FLAG SET WHEN EITHER FIUV, FIU, FIV, FIC ENABLED AND APPROPRIATE EXCEPTION OCCURRED.
14	040000	FID - FLOATING DISABLE INTERRUPTS NO FP INTERRUPTS TO VECTOR 244(8) IF SET.
13, 12		NOT USED
11	004000	FIUV - FLOATING UNDEFINED VARIABLE INTERRUPT IF SET, (-0) MEMORY DATA IS ERROR
10	002000	FIU - FLOATING INTR UNDERFLOW IF SET AND UNDERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY +400(8) IF CLEAR AND UNDERFLOW, ANSWER <-- ZERO
9	001000	FIV - FLOATING OVERFLOW INTERRUPT IF SET AND OVERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY -400(8) IF CLEAR AND OVERFLOW, ANSWER <-- ZERO
8	000400	FIC - FLOATING INTEGER CONVERSION INTERRUPT IF SET AND "STCFI" ERROR, ANSWER <-- ZERO, SET ERROR IF CLEAR AND "STCFI" ERROR, ANSWER <-- ZERO
7	000200	FD - FLOATING MODE 1=DOUBLE, 64 BIT OPERANDS (4W) 0=SINGLE, 32 BIT OPERANDS (2W)
6	000100	FL - INTEGER MODE 1=LONG, 32 BIT INTEGERS (2W) 0=SHORT, 16 BIT INTEGERS (1W)
5	000040	FT - ROUND/TRUNCATE MODE 1=TRUNCATE RESULTS 0=ROUND RESULTS
4	000020	FMM - PUT FP11-E ONLY IN MAINTENANCE MODE
3:0	000017	FN-FZ-FV-FC - FLOATING CONDITION CODES

FLOATING EXCEPTION CODES (FEC):

OCTAL	ENABLE	
00	(NONE)	(NOT USED)
02	(NONE)	FP OPCODE ERROR
04	(NONE)	FP DIVIDE-BY-ZERO ERROR
06	W/FIC	FP INTEGER CONVERSION ERROR
10	W/FIV	FP OVERFLOW ERROR
12	W/FIU	FP UNDERFLOW ERROR
14	W/FIUV	FP UNDEFINED-VARIABLE/(-0) ERROR
16	W/FMM	FP MAINTENANCE TRAP

NOTE - IN "FEC" CODE TYPEOUTS IN ERROR MESSAGES ONLY THE LOW ORDER BYTE IS USED - IGNORE THE PROGRAM FLAG BIT IN THE UPPER BYTE.

```
FLOATING POINT DATA:

IN FLOAT MODE (FD=0), IS 2-16. BIT WORDS, 32. BITS
IN DOUBLE MODE (FD=1), IS 4-16. BIT WORDS, 64. BITS

FIRST WORD: (BOTH F, D MODES)
        B15=SIGN OF NUMBER (1/-, 0/+)
        B14:07=EXPONENT, 8.BITS, FROM -128./+127.
        B06:00=FRACTION, 7.BITS
SECOND WORD: (BOTH F, D MODES)
        B15:00=FRACTION, 16.BITS
THIRD, FOURTH WORDS: (ONLY D MODE)
        B15:00, B15:00=FRACTION, 32. BITS

IN F MODE, THE COMPOSITE 24. BIT FRACTION
IS FORMED BY:

        .1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]

IN D MODE, THE COMPOSITE 56. BIT FRACTION
IS FORMED BY:

        .1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]
                #[WORD3-BIT<15:00>]#[WORD4-BIT<15:00>]

FOR A MORE DETAILED OPERATION/EXPLANATION OF FLOATING POINT
DATA FORMATS AND OPERATIONS, SEE THE PDP-11/6X PROCESSOR
HANDBOOK SECTION ON THE FLOATING POINT INSTRUCTION SET.
```

6.2 RECOVERY

RECOVERY FROM ERRORS HAS BEEN ATTEMPTED TO BE MADE AS AUTOMATIC AND EFFORTLESS AS POSSIBLE. HOWEVER, IN MANY CASES, DUE TO THE NATURE OF THE ERROR, THE PROGRAM MAY NOT EVEN BE ABLE TO BE RUN (EG, IF THE FLOATING POINT MODULE IS IN A HUNG STATE, AND CAN NEVER ENTER THE READY STATE TO ACCEPT A NEW FPP INSTRUCTION). AT THIS POINT, SOLVING THE PROBLEM IS A DIRECT FUNCTION OF THE OPERATORS INGENUITY. THIS TEST SERIES HAS BEEN DESIGNED TO TEST THE FLOATING POINT PROCESSOR SO THAT THESE TYPES OF FAILURES TO RUN WILL BE MINIMAL. THE TESTS HAVE BEEN PLACED IN A SPECIFICALLY STRUCTURED SEQUENCE IN THE PROGRAM TO IMPLEMENT THIS STRATEGY; TESTING THE MOST BASIC ELEMENTS FIRST, PROCEEDING UPWARD IN COMPLEXITY AFTER ESTABLISHING THEIR CORRECT OPERATION. THIS IS WHY IT IS EXTREMELY IMPORTANT THAT THE FLOATING POINT TEST PROGRAMS BE (1) RUN IN THE PRESCRIBED ORDER, AND (2) ONLY BE STARTED AT THEIR BEGINNING ADDRESS (USUALLY 200(8)). THE PROGRAM WILL DISPLAY, AT AN ERROR, THE MOST PERTINENT INFORMATION RELATING TO THE ERROR, AND A BRIEF EXPLANATION OF THE FAILING FUNCTION.

6.3 CAUSES

THESE TEST PROGRAMS ARE NOT HARDWARE ORIENTED, AND AS SUCH IT IS NOT POSSIBLE TO CALL OUT PARTICULAR HARDWARE AREAS AND MODULES RELATING TO A GIVEN FUNCTIONAL FAILURE. HARDWARE DIAGNOSIS FOR A PARTICULAR MACHINE MUST BE DONE USING THE APPROPRIATE ENGINEERING ROM FLOWS AND PRINTS, ALONG WITH THE KNOWN FUNCTIONAL ERRORS (AS DETECTED BY THE PROGRAMS). THIS IS THE INTENT UNDER WHICH THESE INSTRUCTION TESTS WERE DESIGNED AND CODED.

7. RESTRICTIONS

7.1 STARTING

THE PROGRAM MUST BE STARTED AT LOCATION 200(8) ALWAYS.

7.2 OPERATIONAL

THERE ARE NO OPERATIONAL RESTRICTIONS.

8. MISCELLANEOUS

8.1 EXECUTION TIME

MODEL	AVERAGE EXECUTION TIME PER PASS SHORTEST PASS	LONGEST PASS
PDP-11/6X MICROCODE	0:20	9:30
PDP-11/6X W/FP11-E	0:15	5:15

TIMES SPECIFIED AS (MINUTES):(SECONDS)

SHORTEST PASS ::= PASS=1, NO ITERATIONS, USING:
SWR=(004003) FOR PDP-11/6X MICROCODE
SWR=(004002) FOR PDP-11/6X W/FP11-E

LONGEST PASS ::= PASS>=2, 2000. ITERATIONS/TEST, USING:
SWR=(000003) FOR PDP-11/6X MICROCODE
SWR=(000002) FOR PDP-11/6X W/FP11-E

8.2 STACK POINTER

THE STACK POINTER IS SET TO 1100(8) AT THE START OF EACH PASS. IF ALL IS OPERATING CORRECTLY, IT SHOULD ALSO BE THIS VALUE AT

THE START OF EACH TEST, AND AT THE END OF A PASS.

8.3 POWER FAIL

THE TESTS MAY BE POWER FAILED AT ANY TIME. SPURIOUS ERROR MESSAGES MAY OCCUR IF THE FAILURE OCCURRED WHILE THE F.P.U. WAS EXECUTING A FUNCTION, AS NONE OF ITS REGISTERS (FPS, FEC, FEA, ACCUMULATORS) ARE SAVED IN THE EVENT OF A POWER FAILURE. HOWEVER, THESE MESSAGES SHOULD ONLY OCCUR ONCE (IF AT ALL) IMMEDIATELY AFTER POWER IS RESTORED. WHEN POWER IS RESTORED, "POWER" IS TYPED ON THE CONSOLE AND EXECUTION CONTINUES WHERE IT WAS INTERRUPTED.

NOTE THAT THE "VOLATILE" SWITCH REGISTER CONTENTS ARE SAVED AND RESTORED FROM THE STACK IN A POWER FAIL SEQUENCE; THEREFORE THE SWITCH REGISTER SETTINGS SHOULD NOT BE LOST OVER A POWER FAIL.

9. PROGRAM DESCRIPTION

9.1 ORGANIZATION

THESE PROGRAMS ARE ORGANIZED AS MUCH AS POSSIBLE IN A STRAIGHTFORWARD, LINEAR MANNER. THE MAIN BODY OF CODE IS STRUCTURED AS FOLLOWS:

(1) INITIALIZATION ROUTINE
    - SETS UP VECTORS, TYPES HEADER, ETC.
(2) MAIN BODY OF TESTS
    - INLINE TEST CODE, INLINE TEST CALLS
(3) END OF PASS ROUTINE
    - END OF PASS PROCESSING
(4) TEST SUBROUTINES
    - SUBROUTINES CONTAINING COMMON TEST CODE
(5) OVERHEAD ROUTINES
    - SERVICE SUBROUTINES (TYPEOUT,ETC.)

WHEREVER FEASIBLE, COMMON SECIONS OF CODE FOR WIDELY USED FUNCTIONS ARE CONDENSED INTO SUBROUTINES TO CONSERVE MEMORY. THIS INCLUDES NOT ONLY STANDARD SERVICE ROUTINES (SUCH AS SCOPE, ERROR, AND ASCII TYPEOUT), BUT ALSO TESTING ROUTINES WHICH PERFORM VERY SIMILAR FUNCTIONS. THUS IN MANY CASES (THE "ADDF" INSTRUCTION TESTING, FOR EXAMPLE) A SINGLE BODY OF CODE (A SUBROUTINE) IS USED TO PERFORM ALL THE FUNCTIONAL TESTS, WITH A VARIABLE PARAMETER LIST PASSED AT EACH CALL CONTAINING THE DATA OPERANDS AND EXPECTED RESULT FOR EACH INDIVIDUAL TEST. THIS CONSTRUCTION FACILITATES THE ADDITION/DELETION OF TESTS (SHOULD THAT EVER BE NECESSARY), AND ALSO GREATLY CONSERVES MEMORY SPACE REQUIREMENTS WHEN A LARGE NUMBER OF CALLS TO A GIVEN BODY OF CODE ARE REQUIRED.

THE INDIVIDUAL TESTS WITHIN EACH PROGRAM HAVE ALSO BEEN SEQUENCED IN A PARTICULAR ORDER TO FACILITATE THE DETECTION AND RESOLUTION OF ERRORS AS QUICKLY AS POSSIBLE. EACH OF THE TESTS BEGINS AS SIMPLY AS POSSIBLE, FIRST TESTING THE MOST BASIC ELEMENTS. MORE COMPLEX ELEMENTS ARE TESTED AFTERWARDS, EMPLOYING A PHILOSOPHY THAT THE SIMPLER THE TEST, THE BETTER THE RESOLUTION. ALL FUNCTIONS ARE EVENTUALLY TESTED, BUT HOPEFULLY MOST ERRORS WILL BE CAUGHT AND CORRECTED EARLY. A MUCH MORE DETAILED ANALYSIS OF THE SEQUENCE OF TESTS PERFORMED IS PRESENTED IN SECTION 9.2.

9.2 TEST DESCRIPTION

THIS DIAGNOSTIC CONTAINS TESTS FOR THE FLOATING POINT 'ADD-', 'SUB-', 'MUL-', AND 'DIV-' INSTRUCTIONS. ALL COMBINATIONS OF THE SINGLE/DOUBLE, ROUND/TRUNCATE, AND OVERFLOW-UNDERFLOW INTERRUPTS ENABLED/DISABLED MODES ARE EMPLOYED. EACH TEST GENERATES A PAIR OF RANDOM NUMBER OPERANDS, THEN USES BOTH THE HARDWARE AND SOFTWARE ROUTINES TO GENERATE AN ANSWER: EACH SHOULD GENERATE THE SAME ANSWER (WITH A +/- 1 DEVIATION IN THE 'LSB' ALLOWED). FLOATING POINT 'LD-', 'ST-', 'CMP-', AND STATUS INSTRUCTIONS ARE ALSO USED FOR MANIPULATING THE OPERANDS AND RESULTS.

THE PURPOSE OF THESE TESTS IS TO EXERCISE BOTH THE DATA PATH AND CONTROL PORTIONS OF THE FLOATING POINT UNIT SELECTED FOR TESTING WITH AN 'UNLIMITED' SUPPLY OF VARYING OPERANDS, AS MIGHT BE ENCOUNTERED IN A USER/APPLICATION PROGRAM TYPE ENVIRONMENT.

9.3 SUBROUTINE ABSTRACTS

9.3.1 TRAPCATCHER

THE TRAPCATCHER IS A SERIES OF INSTRUCTIONS OCCUPYING THE INTERRUPT VECTOR AREA OF MEMORY. IT CONSISTS OF THE SEQUENCE:

```
        .WORD   .+2     ;PC AFTER TRAP
        .WORD   0       ;PS AFTER TRAP
```

PLACED AT EACH VECTOR ADDRESS IN LOCATIONS 4-776(8) OF MEMORY. THE FIRST WORD OF EACH PAIR ("PC AFTER TRAP") POINTS TO THE SECOND WORD, WHICH SERVES A DUAL PURPOSE AS (1) THE NEW LOADED PS (ALL ZEROS), AND (2) THE NEXT INSTRUCTION TO EXECUTE (0=HALT).

WHEN THE PROGRAM IS EXECUTING, ANY REQUIRED VECTORS ARE SET UP IN THE VECTOR AREA WITH APPROPRIATE VALUES; THE OTHERS BEING LEFT IN THE "TRAPCATCHER" STATE. THUS, IF AN UNEXPECTED TRAP EVER OCCURS IN THE MACHINE, IT WILL BE CAUGHT, AND THE MACHINE SUBSEQUENTLY HALTED, DISPLAYING THE VECTOR ADDRESS + PLUS FOUR

* IN THE ADDRESS LIGHTS.

9.3.2 SCOPE ROUTINE - $SCOPE

THE SCOPE ROUTINE IS ENTERED FROM THE FIRST INSTRUCTION OF EACH TEST IN THE PROGRAM. (NOTE THAT BY DEFINITION, A "TEST" WILL BE DESIGNATED AS THE SECTION OF CODE BETWEEN TWO "SCOPE" STATEMENTS.) THIS ROUTINE PROVIDES THE OVERHEAD CODE NECESSARY TO IMPLEMENT SEVERAL OF THE SWITCH REGISTER CONTROL OPTIONS. UPON ENTRANCE TO A TEST, THE SCOPE STATEMENT AT THE BEGINNING SETS UP CERTAIN LOCATIONS (SEE BELOW) TO SPECIFY THE CURRENT TEST NUMBER AND LOOPING ADDRESS (FOR ITERATIONS). CONTROL IS THEN PASSED TO THE ACTUAL TEST CODE, PERFORMING THE DESIRED TEST. UPON EXIT, THE SCOPE STATEMENT OF THE NEXT TEST IS ENTERED, WHICH DETERMINES WHETHER TO (1) LOOP BACK TO THE PREVIOUS TEST (EG, FOR ITERATIONS) OR (2) INITIALIZE FOR THE NEXT TEST (AS DESCRIBED EARLIER, ABOVE).

ENTRANCE TO THE SCOPE ROUTINE IS VIA AN "IOT" TRAP CALL THROUGH LOCATION 20(8), (FROM THE SCOPE=IOT EQUATE). DEPENDING UPON THE SWITCH SETTINGS (SEE 5.2), CODE IS PRESENT TO: LOAD THE FP11 MICRO BREAK REGISTER, LOOP ON THE CURRENTLY EXECUTING TEST, LOOP ON A SPECIFIC TEST, PERFORM ITERATIONS OF EACH TEST, AND SET UP ADDRESSES FOR POSSIBLE LOOPING ON ERRORS. IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- $MXCNT - MAXIMUM NUMBER OF ITERATIONS PER TEST (GENERALLY WILL BE 2000(10))
- $TSTNM - A COUNTER INDICATING THE NUMBER (1-377(8)) OF THE TEST CURRENTLY BEING EXECUTED
- $LPADR - CONTAINS THE ADDRESS TO WHICH THE SCOPE ROUTINE WILL LOOP, IF THE CURRENT TEST IS BEING LOOPED UPON
- $LPERR - CONTAINS THE ADDRESS TO WHICH THE ERROR ROUTINE (SEE 9.3.3) WILL LOOP, IF AN ERROR OCCURS AND THE LOOPING ON AN ERROR OPTION IS SPECIFIED IN THE SWITCHES. SET UP BY SCOPE, GENERALLY WILL BE THE SAME AS $LPADR, ABOVE.

9.3.3 ERROR ROUTINE - $ERROR

THE ERROR ROUTINE IS ENTERED WHEN THE TEST CODE HAS DETERMINED THAT AN ERROR HAS OCCURRED AS PART OF A TEST. THROUGH USE OF THIS ROUTINE, THE TEST HAS A MEANS OF SIGNALING AN ERROR TO THE 10520 OPERATOR/MONITOR; AND IMPLEMENTING THE CONTROL FUNCTIONS FOR HALTING ON ERROR, BELL ON ERROR, AND LOOPING ON ERROR. IN ADDITION, THE ERROR ROUTINE HAS THE PROVISION TO TYPE OUT ON THE OPERATOR'S CONSOLE A MESSAGE BRIEFLY EXPLAINING THE ERROR, AND SOME OF THE MOST PERTINENT DATA VALUES TO HELP DIAGNOSE THE CAUSE (SEE SECTION 6.2).

THE CALLING MECHANISM IS SIMILAR TO THAT EMPLOYED FOR THE SCOPE ROUTINE (VIA A TRAP), EXCEPT IN THIS INSTANCE, THE "EMT"

INSTRUCTION IS USED, TRAPPING THROUGH LOCATION 30(8). (NOTE THE EQUATE ERROR N=EMT N). THE LOWER BYTE OF THE EMT INSTRUCTION IS CAPABLE OF TRANSMITTING A NUMBER FROM 0-377(8), WHICH WILL BE TERMED THE "ERROR ITEM NUMBER." THIS NUMBER DETERMINES WHICH ERROR MESSAGE, AND ASSOCIATED DATA VALUES WILL BE TYPED OUT WHEN A PARTICULAR ERROR IS SIGNALED. IF THIS NUMBER IS ZERO, JUST THE PC OF THE CALLING "ERROR" INSTRUCTION WILL BE TYPED, OTHERWISE, THE NUMBER IS USED AS AN INDEX THROUGH THE ERROR TABLE ($ERRTB) TO FIND THE APPROPRIATE VALUES TO TYPE (SEE PROGRAM LISTING FOR FURTHER DETAILS).

IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- $REG0 THRU $REG7 - CONTENTS OF GENERAL REGISTERS R0 THRU R7 JUST BEFORE ERROR CALL
- $ERTTL - COMULATIVE NUMBER OF ERRORS ENCOUNTERED TO DATE
- $ERRPC - CONTAINS THE PC OF THE "ERROR" INSTRUCTION JUST EXECUTED
- $LPERR - CONTAINS THE ADDRESS WHICH WILL BE LOOPED UPON FOR THE ERROR LOOPING FACILITY

9.3.4 ERROR MESSAGE TYPEOUT ROUTINE - $TYPERR

THIS ROUTINE ($TYPERR ENTRY POINT) IS CALLED BY THE ERROR PROCESSING ROUTINE DESCRIBED IN 9.3.3 ABOVE. ITS PURPOSE IS TO IMPLEMENT THE ERROR MESSAGE/DATA VALUE ERROR TYPEOUT FACILITY. THE SUBROUTINE WILL, GIVEN THE INDEXING BYTE FROM THE ERROR CALL INSTRUCTION, PICK UP THE CORRECT ERROR MESSAGE VECTOR FROM $ERRTB (ERROR TABLE), AND TYPE OUT THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES ON THE CONSOLE.

9.3.5 TYPE ROUTINE - $TYPE

THIS ROUTINE IS THE STANDARD SYSTEM TYPEOUT ROUTINE FOR ASCII SINGLE-CHARACTER-PER-BYTE STRINGS. IT IS CALLED THROUGH A TRAP INSTRUCTION WITH THE NEXT WORD CONTAINING THE ADDRESS OF THE FIRST CHARACTER IN THE STRING. TYPING TERMINATES WHEN AN ALL-ZERO BYTE IS FOUND. HORIZONTAL TAB STOPS ARE ALSO AUTOMATICALLY PLACED.

9.3.6 OCTAL NUMBER TYPE ROUTINE - $TYPOC

THIS ROUTINE CONVERTS THE TOP NUMBER ON THE STACK TO A 6-DIGIT OCTAL REPRESENTATION, AND TYPES IT ON THE CONSOLE USING THE TYPE ROUTINE $TYPE. SEE LISTING FOR OPTIONS AND FURTHER DETAILS.

9.3.7 POWER UP AND DOWN ROUTINES - $PWRUP AND $PWRDN

THESE TWO ROUTINES ARE ENTERED FOR THE POWER UP AND DOWN CONDITIONS, RESPECTIVELY. THE POWER DOWN ROUTINE ($PWRDN) SAVES THE GENERAL REGISTERS AND STACK POINTER. THE POWER UP

ROUTINE (SPWRUP) CORRESPONDINGLY RESTORES THE REGISTERS, STACK POINTER, AND TYPES THE MESSAGE "POWER" WHEN POWER IS RESTORED. THE VOLATILE INTERNAL SWITCH REGISTER IS ALSO SAVED/RESTORED BY THIS ROUTINE.

9.3.8 END OF PASS ROUTINE - SEOP

THE END OF PASS ROUTINE COUNTS THE NUMBER OF PASSES PERFORMED, DINGS THE BELL/TYPES A MESSAGE (IF ENABLED), SETS/CLEARS THE T-BIT (IF ENABLED), AND ALSO INTERFACES TO THE MONITOR, IF PRESENT. IT ALSO OPTIONALLY LOOPS FOR A NUMBER OF SUBPASSES BEFORE SIGNALLING AN END OF PASS CONDITION.

10. ACT/APT/XXDP

10.1 ACT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE ACT SYSTEM.

10.2 APT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE APT SYSTEM MONITOR. ALL NECESSARY SOFTWARE COMMUNICATION HOOKS ARE PRESENT.

10.3 XXDP COMPATIBILITY

FOR XXDP MEDIA COMPATIBILITY, THE TOP 2K WORDS OF THE 16K WORD MINIMUM MEMORY AREA ARE NOT DISTURBED DURING EXECUTION.

```
                                .TITLE  FPU ADD/SUB/MUL/DIV RANDOM EXER
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DONALD NORTH
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*       SWITCH  OCTAL                   USE
                                ;*       ------  ------          ------------------
                                ;*         15    100000          HALT ON ERROR
                                ;*         14    040000          LOOP ON CURRENTLY EXECUTING TEST
                                ;*         13    020000          INHIBIT ERROR TYPEOUTS
                                ;*         12    010000          INHIBIT STATUS TYPEOUTS
                                ;*         11    004000          INHIBIT ITERATIONS
                                ;*         10    002000          0=BELL ON PASS END
                                ;*               002000          1=BELL ON ERROR
                                ;*          9    001000          LOOP ON ERROR
                                ;*          8    000400          LOOP ON TEST NUMBER IN "$LPTST"
                                ;*          1    000000          0=TEST HFP/WFP ALTERNATELY EACH PASS
                                ;*               000002          1=TEST ONLY UNIT SPECIFIED IN SW<00>
                                ;*          0    000002          0=SELECT HFP, IF SW<01>=1
                                ;*               000001          1=SELECT WFP, IF SW<01>=1
                                
                                .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 INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                                
                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
```



```
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER
                                
                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7
                                
                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
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
```

```
                        ;BASIC DEFINITIONS

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

                        ;*MED INSTR CODES
076600                  MED=    076600          ;OPCODE

000022                  RWHAM1= 022             ;READ WHAM1

000144                  RFLAG=  144             ;READ FLAGS
000344                  WFLAG=  344             ;WRITE FLAGS

                        ;*FLOATING POINT INTERRUPT VECTOR
000244                  FPPVEC= 244

                        ;*FLOATING POINT REGISTER DEFINITIONS
000000                  AC0=    %0
000001                  AC1=    %1
000002                  AC2=    %2
000003                  AC3=    %3
000004                  AC4=    %4
000005                  AC5=    %5



                        .SBTTL  TRAP CATCHER

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



```
000174  000000          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  003000          JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

                        .SBTTL  ACT11 HOOKS

                        ;;*****************************************************************
                        ;HOOKS REQUIRED BY ACT11
000204                          $SVPC=.                 ;SAVE PC
000046                          .=46
000046  072106                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
000052                          .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
003204                          .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                        .SBTTL  APT PARAMETER BLOCK

                        ;;*****************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                        ;;*****************************************************************
001000                          .SX=.   ;;SAVE CURRENT LOCATION
000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000200                          200     ;;FOR APT START UP
000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
001000  001000                  $APTHDR ;;POINT TO APT HEADER BLOCK
001000                          .=.SX   ;;RESET LOCATION COUNTER
                        ;;*****************************************************************
                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNUSTIC
                        ;INTERFACE SPEC.

001000                  $APTHD:
001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001202          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000012          $TSTM:  .WORD   10.     ;;RUN TIM OF LONGEST TEST
001006  000055          $PASTM: .WORD   45.     ;;RUN TIME IN SECS, OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000          $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000014                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
                                ;;-----START OF CLEAR COMMON TAGS-----
001100  000000                  .WORD   0
001102  000000          $ISTNM: .WORD   0               ;;CONTAINS THE TEST NUMBER
001104  000000          $ERFLG: .WORD   0               ;;CONTAINS ERROR FLAG
001106  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001110  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001112  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001114  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001116  000000          $ITEMB: .WORD   0               ;;CONTAINS ITEM CONTROL BYTE
001120  000001          $ERMAX: .WORD   1               ;;CONTAINS MAX. ERRORS PER TEST
001122  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001124  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF "GOOD" DATA
001126  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF "BAD" DATA
001130  000000          $GDDAT: .WORD   0               ;;CONTAINS "GOOD" DATA
001132  000000          $BDDAT: .WORD   0               ;;CONTAINS "BAD" DATA
001134  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001136  000000                  .WORD   0
001140  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001141  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001142  000000                  .WORD   0
                                ;;-----END OF CLEAR COMMON TAGS-----
001144  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001146  177570          DISPLA: .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
001150  000000          $LPTST: .WORD   0               ;;CONTAINS TEST NUMBER TO LOOP UPON
001152  177560          $TKS:   177560                  ;;TTY KBD STATUS
001154  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001156  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001160  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001162     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001163     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001164     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001165     000          $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001166  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001170  000003          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001172  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001176     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001177     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001200  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

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



```
001212  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001214  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001216  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001220  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001222                  $ETABLE:                ;;APT ENVIRONMENT TABLE
001222     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001223     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001224  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001226  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001230  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001232                          $ETEND:
                                .MEXIT
```

```
                                        .SBTTL  ERROR POINTER TABLE

001232                                  $ERRTB:

                                        ;*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 (SERRPC).
                                        ;*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
                                                ;*NOTE:  ERROR VECTOR TABLE ($ERRTB) HAS BEEN MODIFIED,
                                                ;*       SEE $ERRTYP ROUTINE FOR ITS STRUCTURE

001232  032524  033204  033474  EMV001: .WORD   EMJ,DHA,DTA,0,0,0,0,0,0,0        ;ADDF   *
001240  000000  000000  000000
001246  000000  000000  000000
001254  000000
001256  032545  033204  033474  EMV002: .WORD   EMK,DHA,DTA,0,0,0,0,0,0,0        ;SUBF   * FPS ERRORS
001264  000000  000000  000000
001272  000000  000000  000000
001300  000000
001302  032566  033204  033474  EMV003: .WORD   EML,DHA,DTA,0,0,0,0,0,0,0        ;MULF   *
001310  000000  000000  000000
001316  000000  000000  000000
001324  000000
001326  000000  000000  000000  EMV004: .WORD   0,0,0,0,0,0,0,0,0,0              ;(UNUSED)
001334  000000  000000  000000
001342  000000  000000  000000
001350  000000
001352  032275  033220  033514  EMV005: .WORD   EME,DHB,DTD,LOF,HIF,0,0,0,0,0    ;ADDF   *
001360  033622  033632  000000
001366  000000  000000  000000
001374  000000
001376  032275  033256  033526  EMV006: .WORD   EME,DHC,DTE,LOD,HID,0,0,0,0,0    ;ADDD   *
001404  033572  033606  000000
001412  000000  000000  000000
001420  000000
001422  032325  033220  033514  EMV007: .WORD   EMF,DHB,DTD,LOF,HIF,0,0,0,0,0    ;SUBF   *
001430  033622  033632  000000
001436  000000  000000  000000
001444  000000
001446  032325  033256  033526  EMV010: .WORD   EMF,DHC,DTE,LOD,HID,0,0,0,0,0    ;SUBD   * RESULT ERRORS
001454  033572  033606  000000
001462  000000  000000  000000
001470  000000
001472  032355  033220  033514  EMV011: .WORD   EMG,DHB,DTD,LOF,HIF,0,0,0,0,0    ;MULF   *
001500  033622  033632  000000
001506  000000  000000  000000
001514  000000
001516  032355  033256  033526  EMV012: .WORD   EMG,DHC,DTE,LOD,HID,0,0,0,0,0    ;MULD   *
001524  033572  033606  000000
```



```
001532  000000  000000  000000
001540  000000
001542  032405  033220  033514  EMV013: .WORD   EMH,DHB,DTD,LOF,HIF,0,0,0,0,0    ;DIVF   *
001550  033622  033632  000000
001556  000000  000000  000000
001564  000000
001566  032405  033256  033526  EMV014: .WORD   EMH,DHC,DTE,LOD,HID,0,0,0,0,0    ;DIVD   *
001574  033572  033606  000000
001602  000000  000000  000000
001610  000000

001612  032435  033354  033550  EMV015: .WORD   EMI,DHD,DTF,0,0,0,0,0,0,0        ;ILLEGAL FPP TRAP
001620  000000  000000  000000
001626  000000  000000  000000
001634  000000

001636  032607  033204  033474  EMV016: .WORD   EMM,DHA,DTA,0,0,0,0,0,0,0        ;DIVF   *
001644  000000  000000  000000
001652  000000  000000  000000
001660  000000
001662  032630  033204  033474  EMV017: .WORD   EMN,DHA,DTA,0,0,0,0,0,0,0        ;ADDD   *
001670  000000  000000  000000
001676  000000  000000  000000
001704  000000
001706  032651  033204  033474  EMV020: .WORD   EMO,DHA,DTA,0,0,0,0,0,0,0        ;SUBD   * FPS ERRORS
001714  000000  000000  000000
001722  000000  000000  000000
001730  000000
001732  032672  033204  033474  EMV021: .WORD   EMP,DHA,DTA,0,0,0,0,0,0,0        ;MULD   *
001740  000000  000000  000000
001746  000000  000000  000000
001754  000000
001756  032713  033204  033474  EMV022: .WORD   EMQ,DHA,DTA,0,0,0,0,0,0,0        ;DIVD   *
001764  000000  000000  000000
001772  000000  000000  000000
002000  000000

002002  032734  033433  033502  EMV023: .WORD   EMR,DHE,DTB,0,0,0,0,0,0,0        ;ADDF   *
002010  000000  000000  000000
002016  000000  000000  000000
002024  000000
002026  032761  033433  033502  EMV024: .WORD   EMS,DHE,DTB,0,0,0,0,0,0,0        ;SUBF   *
002034  000000  000000  000000
002042  000000  000000  000000
002050  000000
002052  033026  033433  033502  EMV025: .WORD   EMT,DHE,DTB,0,0,0,0,0,0,0        ;MULF   *
002060  000000  000000  000000
002066  000000  000000  000000
002074  000000
002076  033033  033433  033502  EMV026: .WORD   EMU,DHE,DTB,0,0,0,0,0,0,0        ;DIVF   * FEC/FEA ERRORS
002104  000000  000000  000000
002112  000000  000000  000000
002120  000000
002122  033060  033433  033502  EMV027: .WORD   EMV,DHE,DTB,0,0,0,0,0,0,0        ;ADDD   *
002130  000000  000000  000000
002136  000000  000000  000000
```

```
394  002144  000000
395  002146  033105  033433  033507  EMV030: .WORD   EMW,DHE,DTB,0,0,0,0,0,0,0        ;SUBD   *
396  002154  000000  000000  000000
397  002162  000000  000000  000000
398  002170  000000
399  002172  033132  033433  033502  EMV031: .WORD   EMX,DHE,DTB,0,0,0,0,0,0,0        ;MULD   *
400  002200  000000  000000  000000
401  002206  000000  000000  000000
402  002214  000000
403  002216  033157  033433  033502  EMV032: .WORD   EMY,DHE,DTB,0,0,0,0,0,0,0        ;DIVD   *
404  002224  000000  000000  000000
405  002232  000000  000000  000000
406  002240  000000
407
408  002242  032103  033204  033474  EMV033: .WORD   EMA,DHA,DTA,0,0,0,0,0,0,0        ;F-MODE EXERCISER  *  FPS
409  002250  000000  000000  000000
410  002256  000000  000000  000000
411  002264  000000
412  002266  032140  033204  033474  EMV034: .WORD   EMB,DHA,DTA,0,0,0,0,0,0,0        ;D-MODE EXERCISER  *    ERROR
413  002274  000000  000000  000000
414  002302  000000  000000  000000
415  002310  000000
416
417  002312  032175  033220  033514  EMV035: .WORD   EMC,DHB,DTD,OP1F,OP2F,OP3F,OP4F,OP5F,OP6F-0      ;F-MODE EXERCISER  *  RE
                                                                                                          SULT
418  002320  033642  033652  033662
419  002326  033672  033702  033712
420  002334  000000
421  002336  032235  033256  033526  EMV036: .WORD   EMD,DHC,DTF,OP1D,OP2D,OP3D,OP4D,OP5D,OP6D,0      ;D-MODE EXERCISER  *
                                                                                                          ERROR
422  002344  033722  033736  033752
423  002352  033766  034002  034016
424  002360  000000
425
426
427
428                                  ,SBTTL  PROGRAM DEFINED COMMON TAGS
429                                  ;*VARIABLES
430  002362  000000                  FPS:    .WORD   0               ;FPS STORED HERE AFTER STFPS
431  002364  000000                  FEC:    .WORD   0               ;FEC STORED HERE AFTER STST
432  002366  000000                  FEA:    .WORD   0               ;FEA STORED HERE AFTER STST
433  002370  000000                  FPPOPC: .WORD   0               ;OLD PC SAVED HERE AFTER TRAP
434  002372  000000                  FPPOPS: .WORD   0               ;OLD PS SAVED HERE AFTER TRAP
435  002374  000000                  FPPOSP: .WORD   0               ;SP AFTER TRAP
436  002376  000000                  EXPFEA: .WORD   0               ;EXPECTED FEA
437  002400  000000                  $FPS:   .WORD   0               ;SOFTWARE FPS
438  002402  000000                  $FEC:   .WORD   0               ;SOFTWARE FEC
439  002404  000000                  $FEA:   .WORD   0               ;SOFTWARE FEA
440  002406  000000  000000  000000  ANS1:   .WORD   0,0,0,0         ;HARDWARE FLOATING POINT ANSWER
441  002414  000000
442  002416  000000  000000  000000  ANS2:   .WORD   0,0,0,0         ;SOFTWARE FLOATING POINT ANSWER
443  002424  000000
444  002426  027005  104552  111730  LONUM:  .WORD   027005,104552,111730,000555    ;RANDOM EXERCISER OPERAND LONUM
445  002434  000555
446  002436  102337  166330  007025  HINUM:  .WORD   102337,166330,007025,021553    ;RANDOM EXERCISER OPERAND HINUM
447  002444  021553
```



```
448  002446  175463  030712  105726  OP1:    .WORD   175463,030712,105726,124064    ;
449  002454  124064
450  002456  156007  002361  070707  OP2:    .WORD   156007,002361,070707,061111    ;
451  002464  061111
452  002466  003505  134261  062451  OP3:    .WORD   003505,134261,062451,052525    ;EXTRA RANDOM OPERANDS
453  002474  052525
454  002476  102466  123456  111111  OP4:    .WORD   102466,123456,111111,101010    ;
455  002504  101010
456  002506  076103  000346  000612  OP5:    .WORD   076103,000346,000612,125252    ;
457  002514  125252
458  002516  044100  112400  177777  OP6:    .WORD   044100,112400,177777,000006    ;
459  002524  000006
460
461                                  ;*ADDED COUNTERS FOR TRACING WHERE WE'VE BEEN
462  002526  000000                  ADDC0:  .WORD   0               ;TOTAL NUMBER           *
463  002530  000000                  ADDC1:  .WORD   0               ;A=0, B#0               *
464  002532  000000                  ADDC2:  .WORD   0               ;A#0, B=0               *
465  002534  000000                  ADDC3:  .WORD   0               ;A=0, B=0               *  FOR
466  002536  000000                  ADDC4:  .WORD   0               ;NO SHIFT REQ'D         *  ADD
467  002540  000000                  ADDC5:  .WORD   0               ;OVERFLOW               *  AND
468  002542  000000                  ADDC6:  .WORD   0               ;OVERFLOW, ENABLED      *  SUBTRACT
469  002544  000000                  ADDC7:  .WORD   0               ;UNDERFLOW              *
470  002546  000000                  ADDC8:  .WORD   0               ;UNDERFLOW, ENABLED     *
471
472  002550  000000                  MULC0:  .WORD   0               ;TOTAL NUMBER           *
473  002552  000000                  MULC1:  .WORD   0               ;A=0 AND/OR B=0         *
474  002554  000000                  MULC2:  .WORD   0               ;OVERFLOW               *  FOR
475  002556  000000                  MULC3:  .WORD   0               ;OVERFLOW, ENABLED      *  MULTIPLY
476  002560  000000                  MULC4:  .WORD   0               ;UNDERFLOW              *
477  002562  000000                  MULC5:  .WORD   0               ;UNDERFLOW, ENABLED     *
478
479  002564  000000                  DIVC0:  .WORD   0               ;TOTAL NUMBER           *
480  002566  000000                  DIVC1:  .WORD   0               ;NUM A=0                *
481  002570  000000                  DIVC2:  .WORD   0               ;DEN B=0                *  FOR
482  002572  000000                  DIVC3:  .WORD   0               ;OVERFLOW               *  DIVIDE
483  002574  000000                  DIVC4:  .WORD   0               ;OVERFLOW, ENABLED      *
484  002576  000000                  DIVC5:  .WORD   0               ;UNDERFLOW              *
485  002600  000000                  DIVC6:  .WORD   0               ;UNDERFLOW, ENABLED     *
486
487                                  ;*REGISTER CONTENTS, AT ERROR, STORED HERE
488  002602  000000                  EREG0:  .WORD   0
489  002604  000000                  EREG1:  .WORD   0
490  002606  000000                  EREG2:  .WORD   0
491  002610  000000                  EREG3:  .WORD   0
492  002612  000000                  EREG4:  .WORD   0
493  002614  000000                  EREG5:  .WORD   0
494  002616  000000                  EREG6:  .WORD   0
495  002620  000000                  EREG7:  .WORD   0
496
497                                  ;*MESSAGES FOR BEGIN PROGRAM/START OF PASS
498  002622  005015  005012  042115  BGNMES: .ASCII  <CR><LF><LF><LF>"MP-11-DQFPD-"
499  002630  038455  026461  050504
500  002636  050106  026504
501  002642  027102                          .ASCII  "B,"
502  002644  027056                          .ASCII  ".."
503  002646  042120  026520  030461          .ASCII  "PDP-11/6X F.P.U. ADD/SUB/MUL/DIV EXERCISER"<CR><LF>
```

```
   504  002654  033057  020130  027106
   505  002662  027120  027125  040440
   506  002670  042104  051457  041125
   507  002676  046457  046125  042057
   508  002704  053111  042440  042530
   509  002712  041522  051511  051105
   510  002720  005015     000
   511  002723     015  050012  051501  NWPAS1: .ASCIZ  <CR><LF>"PASS #"
   512  002730  020123  000043
```



```
   513                                  .SBTTL  START OF PASS ROUTINE
   514
   515
   516                                  ;!*****************************************************************
   517                                          .ENABL  AMA             ; ASSEMBLE ALL RELATIVE REFERENCES AS ABSOLUTE
   518                                  ;!*****************************************************************
   519
   520  003000                          START:
   521                                  .SBTTL  INITIALIZE THE COMMON TAGS
   522                                  ;;CLEAR THE COMMON TAGS (#CMTAG) AREA
   523  003000  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
   524  003004  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
   525  003006  022706  001144                  CMP     #SWR,R6 ;;DONE?
   526  003012  001374                          BNE     .-6             ;;LOOP BACK IF NO
   527  003014  012706  001100                  MOV     #STACK,SP       ;!SETUP THE STACK POINTER
   528                                  ;!INITIALIZE A FEW VECTORS
   529  003020  012737  027672  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
   530  003026  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
   531  003034  012737  030150  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
   532  003042  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
   533  003050  012737  031630  000034          MOV     #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
   534  003056  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
   535  003064  012737  031676  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
   536  003072  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
   537  003100  013737  022042  022034          MOV     @ENDCT,$LOPCT   ;;SETUP END-OF-PROGRAM COUNTER
   538  003106  005037  001166                  CLR     @TIMES          ;!INITIALIZE NUMBER OF ITERATIONS
   539  003112  005037  001170                  CLR     @ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
   540  003116  012737  000001  001120          MOV     #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
   541  003124  012737  003124  001110          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
   542  003132  012737  003132  001112          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
   543                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
   544                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
   545  003140  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
   546  003144  012737  003200  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
   547  003152  012737  177570  001144          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWITCH REGISTER
   548  003160  012737  177570  001146          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
   549  003166  022777  177777  175750          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
   550  003174  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
   551                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
   552  003176  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
   553  003200  012716  003206          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
   554  003204  000002                          RTI
   555  003206  012737  000176  001144  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
   556  003214  012737  000174  001146          MOV     #DISPREG,DISPLAY
   557  003222  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
   558
   559  003226  005037  001210                  CLR     @PASS           ;;CLEAR PASS COUNT
   560  003232  132737  000200  001223          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
   561  003240  001403                          BEQ     67$             ;!YES,USE NON-APT SWITCH
   562  003242  012737  001224  001144          MOV     #$SWREG,SWR     ;!NO,USE APT SWITCH REGISTER
   563  003250                          67$:
   564
   565                                          ;SET UP FPP UNEXPECTED TRAP CATCHER * * * * * * * *
   566  003250  012737  023256  000244          MOV     #FPPILT,@#FPPVEC        ;NEW PC AT FPP TRAP
   567  003256  005037  000246                  CLR     @#FPPVEC+2              ;NEW PS AT FPP TRAP
   568
```

```
                                        ;*CLEAR COUNTERS FOR TRACING INFO
003262  012700  002526                  MOV     #ADDC0,R0       ;FIRST LOCATION TO CLEAR
003266  012701  000026                  MOV     #26,R1          ;26(8) WONDS
003272  005020                  26$:    CLR     (R0)+           ;CLEAR AND BUMP
003274  077102                          SOB     R1,26$          ;DO AGAIN

003276  104401  002622                  TYPE    ,BGNMES         ;ID MESSAGE AT START

                                ;////////////////////////////////////////////////////////////////////////
                                ;       MESSAGE ON WHETHER OR NOT NFP UNIT IS PRESENT

003302  076600  000022                  MFD     ,RWHAMI                 ;WHAMI INTO R0
003306  032700  000020                  BIT     #BIT04,R0               ;IS THERE A NFP UNIT ?
003312  001403                          BEQ     70$                     ;NO, BR
003314  104401  003330                  TYPE    ,68$                    ;INDICATE FP11-E PRESENT
003320  000453                          BR      NEWPAS                  ;GO FOR SUBPASS INIT
003322  104401  003370          70$:    TYPE    ,69$                    ;INDICATE NO FP11-E
003326  000450                          BR      NEWPAS                  ;GO FOR SUBPASS INIT

003330  005015  020052  050106  68$:    .ASCIZ  <15><12>"* FP11-E NFP UNIT PRESENT *"<15><12>
003336  030461  042455  044040
003344  050106  052440  044516
003352  020124  051120  051505
003360  047105  020124  006452
003366  000012
003370  005015  020052  047516  69$:    .ASCIZ  <15><12>"* NO FP11-E NFP UNIT - ALL TESTS WFP ONLY *"<15><12>
003376  040040  030520  026461
003404  020105  043110  020120
003412  047125  052111  026440
003420  040440  046114  052040
003426  051505  051524  053440
003434  050106  047440  046116
003442  020131  006452  000012
                        .EVEN

                                ;////////////////////////////////////////////////////////////////////////

                                ;;**********************************************************
                                ;;      NEW PASS ENTERS HERE
                                ;;**********************************************************

003450  012706  001100  NEWPAS: MOV     #STACK,SP               ;RESET STACK PTR

003454  032777  010000  175462          BIT     #BIT12,@SWR             ;INHIBIT STATUS TYPEOUTS ?
003462  001015                          BNE     SUBPAS                  ;BR IF YES
003464  023737  022042  022034          CMP     $ENDCT,$EOPCT           ;TIME FOR A MESSAGE ?
003472  001011                          BNE     SUBPAS                  ;NO, NOT YET

003474  104401  002723                  TYPE    ,NWPAS1                 ;"PASS #"
003500  013746  001210                  MOV     @PASS,-(SP)             ;PASS COUNT INTO ...
003504  005216                          INC     (SP)                    ;  1-N RANGE
003506  104403                          TYPOS                           ;TYPE OCTAL
003510     006     000                  .BYTE   6,0                     ;  6 DIGITS, NO LEADING ZEROS
003512  104401  001177                  TYPE    ,$CRLF                  ;END THE LINE
```



```
                                ;;**********************************************************
                                ;;      NEW SUBPASS ENTERS HERE
                                ;;**********************************************************

003516  012706  001100  SUBPAS: MOV     #STACK,SP               ;RESET SP FOR INSURANCE

003522  076600  000022          MFD     ,RWHAMI                 ;GET WHAMI INTO R0
003526  032700  000020          BIT     #BIT04,R0               ;1=NFP PRESENT, 0=NO
003532  001430                  BEQ     20$                     ;IF NO NFP, TEST WARM ONLY

003534  076600  000144          MFD     ,NFLAG                  ;GET FLAGS INTO R0

003540  032777  000002  175376          BIT     #SW01,@SWR              ;SW01:  1=NFP OR NFP TEST ONLY
003546  001411                          BEQ     1$                      ;       0=ALTERNATE NFP/WFP PER PASS

003550  032777  000001  175366          BIT     #SW00,@SWR              ;SW00:  1=WFP ONLY
003556  001403                          BEQ     2$                      ;       0=NFP ONLY
003560  042700  001000                  BIC     #BIT12,R0               ;CLEAR NFP ENABLE FLAG<5> FOR WFP
003564  000402                          BR      3$                      ;
003566  052700  010000          2$:     BIS     #BIT12,R0               ;SET NFP ENABLE FLAG<5> FOR NFP
003572  076600  000344          3$:     MFD     ,NFLAG                  ;REWRITE FLAGS

003576  032700  010000          1$:     BIT     #BIT12,R0               ;TEST WHO'S ENABLED: HOT, WARM
003602  001404                          BEQ     2W$                     ;SET APPROPRIATE HEADER:

003604  012737  032066  030414  19$:    MOV     #ASCHOT,HOTWRM          ;"HOT: "
003612  000403                          BR      21$                     ;
003614  012737  032074  030414  20$:    MOV     #ASCWRM,HOTWRM          ;"WARM: "
003622                          21$:
```

```
                                ;;*****************************************************************
                                ;*TEST 1         EXERCISE ADDF, ALL INTERRUPTS ON, ROUNDING MODE
                                ;;*****************************************************************
003622  000004          TST1:   SCOPE
003624  012737  003756  002376          MOV     #ARET1,EXPFEA   ;ADDR OF INSTR BEING TESTED
003632  012737  007400  002400          MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
003640  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
003644  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
003650  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
003654  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
003660  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
003664  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
003670  002426  002436                  .WORD   LONUM,HINUM     ;
003674  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
003700  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
003704  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
003710  023566                          $ADD                    ;ADDRESS OF FORTRAN ADD
003712  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                ;
003716  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
003722  170127  040000                  LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
003726  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
003732  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
003736  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
003742  170127  007400                  LDFPS   #007400         ;TURN INTERRUPTS ON
003746  012737  003754  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                ;*****************************************************************
                                ;
003754  172600                  ARET1:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
003756  172201                          ADDF    AC1,AC2         ;ADD AC1 BY AC2
003760  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003764  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
003772  001403                          BEQ     AERR1           ;BRANCH IF OK
003774  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004000  104001                          ERROR   1               ;FPS ERROR
                                ;
004002  005737  002400          AERR1:  TST     $FPS            ;ERROR BIT SET ?
004006  100014                          BPL     ATST1           ;NO, DONT GET FEC/FEA
004010  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                ;
004014  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
004022  001401                          BEQ     1$              ;BRANCH IF OK
004024  104023                          ERROR   23              ;FEC IS WRONG
                                ;
004026  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
004034  001401                          BEQ     ATST1           ;BRANCH IF OK
004036  104023                          ERROR   23              ;WRONG ADDRESS IN FEA
                                ;
004040  173702                  ATST1:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
004042  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
004044  001416                          BEQ     AEND1           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
004046  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004052  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
004060  005637  002406                  SBC     ANS1
```



```
004064  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
004070  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
004072  001403                          BEQ     AEND1           ;BRANCH IF OK
004074  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004100  104005                          ERROR   5               ;FPU AND FORTRAN DISAGREE
                                ;
004102  005037  002362          AEND1:  CLR     FPS             ;CLR FPU FPS BUFFER
                                ;
                                ;
                                ;;*****************************************************************
                                ;*TEST 2         EXERCISE ADDD, ALL INTERRUPTS ON, ROUNDING MODE
                                ;;*****************************************************************
004106  000004          TST2:   SCOPE
004110  012737  004242  002376          MOV     #ARET2,EXPFEA   ;ADDR OF INSTR BEING TESTED
004116  012737  007600  002400          MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
004124  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
004130  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
004134  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
004140  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
004144  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
004150  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
004154  002426  002436                  .WORD   LONUM,HINUM     ;
004160  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
004164  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
004170  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
004174  023566                          $ADD                    ;ADDRESS OF FORTRAN ADD
004176  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
                                ;
004202  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
004206  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
004212  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
004216  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
004222  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
004226  170127  007600                  LDFPS   #007600         ;TURN INTERRUPTS ON
004232  012737  004240  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                ;*****************************************************************
                                ;
004240  172600                  ARET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
004242  172201                          ADDD    AC1,AC2         ;ADD AC1 BY AC2
004244  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
004250  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
004256  001403                          BEQ     AERR2           ;BRANCH IF OK
004260  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
004264  104017                          ERROR   17              ;FPS ERROR
                                ;
004266  005737  002400          AERR2:  TST     $FPS            ;ERROR BIT SET ?
004272  100014                          BPL     ATST2           ;NO, DONT GET FEC/FEA
004274  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                ;
004300  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
004306  001401                          BEQ     1$              ;BRANCH IF OK
004310  104027                          ERROR   27              ;FEC IS WRONG
                                ;
004312  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
```

```
767  004320  001401                        BEQ     ATST2           ;BRANCH IF OK
768  004322  104027                        ERROR   27              ;WRONG ADDRESS IN FEA
769
770  004324  173702                ATST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
771  004326  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
772  004330  001422                        BEQ     AEND2           ;ANSWERS CHECK
773                                         ;COMPENSATE FOR FORTRAN INACCURACIES.
774  004332  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
775  004336  162737  000001  002414        SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
776  004344  005637  002412                SBC     ANS1+4
777  004350  005637  002410                SBC     ANS1+2
778  004354  005637  002406                SBC     ANS1
779  004360  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
780  004364  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
781  004366  001403                        BEQ     AEND2           ;BRANCH IF OK
782  004370  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
783  004374  104006                        ERROR   6               ;FPU AND FORTRAN DISAGREE
784
785  004376  005037  002362        AEND2:  CLR     FPS             ;CLR FPU FPS BUFFER
786
787
788
789
790                                 ;;*****************************************************
791                                 ;*TEST 3         EXERCISE ADDF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
792                                 ;;*****************************************************
793  004402  000004                TST3:   SCOPE
794  004404  012737  004536  002376        MOV     #ARET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
795  004412  012737  004400  002400        MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
796  004420  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
797  004424  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
798  004430  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
799  004434  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
800  004440  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
801  004444  004737  023342                JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
802  004450  002426  002436                .WORD   LONUM,HINUM     ;
803  004454  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
804  004460  023510  002426                $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
805  004464  023510  002436                $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
806  004470  023566                        $ADD                    ;ADDRESS OF FORTRAN ADD
807  004472  023540  002416                $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
808
809  004476  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
810  004502  170127  040000                LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
811  004506  172437  002426                LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
812  004512  172537  002436                LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
813  004516  172737  002416                LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
814  004522  170127  004400                LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
815  004526  012737  004534  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
816
817                                         ;*************************************************
818
819  004534  172600                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
820  004536  172701                ARET3:  ADDF    AC1,AC2         ;ADD AC1 BY AC2
821  004540  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
822  004544  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
```



```
823  004552  001403                        BEQ     ATST3           ;BRANCH IF OK
824  004554  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
825  004560  104001                        ERROR   1               ;FPS ERROR
826
827  004562  173702                ATST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
828  004564  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
829  004566  001416                        BEQ     AEND3           ;ANSWERS CHECK
830                                         ;COMPENSATE FOR FORTRAN INACCURACIES.
831  004570  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
832  004574  162737  000001  002410        SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
833  004602  005637  002406                SBC     ANS1
834  004606  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
835  004612  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
836  004614  001403                        BEQ     AEND3           ;BRANCH IF OK
837  004616  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
838  004622  104005                        ERROR   5               ;FPU AND FORTRAN DISAGREE
839
840  004624  005037  002362        AEND3:  CLR     FPS             ;CLR FPU FPS BUFFER
841
842
843
844
845                                 ;;*************************************************************
846                                 ;*TEST 4         EXERCISE ADDD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
847                                 ;;*************************************************************
848  004630  000004                TST4:   SCOPE
849  004632  012737  004764  002376        MOV     #ARET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
850  004640  012737  004600  002400        MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
851  004646  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
852  004652  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
853  004656  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
854  004662  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
855  004666  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
856  004672  004737  023332                JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
857  004676  002426  002436                .WORD   LONUM,HINUM     ;
858  004702  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
859  004706  023510  002426                $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
860  004712  023510  002436                $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
861  004716  023566                        $ADD                    ;ADDRESS OF FORTRAN ADD
862  004720  023540  002416                $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
863
864  004724  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
865  004730  170127  040200                LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
866  004734  172437  002426                LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
867  004740  172537  002436                LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
868  004744  172737  002416                LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
869  004750  170127  004600                LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
870  004754  012737  004762  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
871
872                                         ;*************************************************************
873
874  004762  172600                        LDD     AC0,AC2         ;LOAD AC0 INTO AC2
875  004764  172201                ARET4:  ADDD    AC1,AC2         ;ADD AC1 BY AC2
876  004766  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
877  004772  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
878  005000  001403                        BEQ     ATST4           ;BRANCH IF OK
```

```
000879  005002  174237  002406              STD     AC2,ANS1        ;SAVE FPU ANSWER
000880  005006  104017                      ERROR   17              ;FPS ERROR
000881
000882  005010  173702              ATST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
000883  005012  170000                      CFCC                    ;COPY FLOATING CONDITION CODES
000884  005014  001422                      BEQ     AEND4           ;ANSWERS CHECK
000885                                      ;COMPENSATE FOR FORTRAN INACCURACIES.
000886  005016  174237  002406              STD     AC2,ANS1        ;SAVE FPU ANSWER
000887  005022  162737  000001  002414      SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
000888  005030  005637  002412              SBC     ANS1+4
000889  005034  005637  002410              SBC     ANS1+2
000890  005040  005637  002406              SBC     ANS1
000891  005044  173737  002406              CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
000892  005050  170000                      CFCC                    ;COPY FLOATING CONDITION CODES
000893  005052  001403                      BEQ     AEND4           ;BRANCH IF OK
000894  005054  174237  002406              STD     AC2,ANS1        ;SAVE FPU ANSWER
000895  005060  104006                      EPROR   6               ;FPU AND FORTRAN DISAGREE
000896
000897  005062  005037  002362      AEND4:  CLR     FPS             ;CLR FPU FPS BUFFER
000898
000899
000900
000901
000902                                      ;;*****************************************************************
000903                                      ;*TEST 5         EXERCISE ADDF, ALL INTERRUPTS ON, TRUNCATE MODE
000904                                      ;;*****************************************************************
000905  005066  000004              TST5:   SCOPE
000906  005070  012737  005222  002376      MOV     #ARET5,EXPFEA   ;ADDR OF INSTR BEING TESTED
000907  005076  012737  007440  002400      MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
000908  005104  005037  002402              CLR     $FEC            ;CLR FORTRAN FEC
000909  005110  005037  002404              CLR     $FEA            ;CLR FORTRAN FEA
000910  005114  005037  002362              CLR     FPS             ;CLR FPU FPS BUFFER
000911  005120  005037  002364              CLR     FEC             ;CLR FPU FEC BUFFER
000912  005124  005037  002366              CLR     FEA             ;CLR FPU FEA BUFFER
000913  005130  004737  023342              JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
000914  005134  002426  002436              .WORD   LONUM,HINUM     ;
000915  005140  004437  023506              JSR     R4,$POLSH       ;ENTER POLISH MODE
000916  005144  023510  002426              $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
000917  005150  023510  002436              $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
000918  005154  023566                      $ADD                    ;ADDRESS OF FORTRAN ADD
000919  005156  023540  002416              $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
000920
000921  005162  013700  002400              MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
000922  005166  170127  040000              LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
000923  005172  172437  002426              LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
000924  005176  172537  002436              LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
000925  005202  172737  002416              LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
000926  005206  170127  007440              LDFPS   #007440         ;TURN INTERRUPTS ON
000927  005212  012737  005220  001110      MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
000928
000929                                      ;*****************************************************************
000930
000931  005220  172600                      LDF     AC0,AC2         ;LOAD AC0 INTO AC2
000932  005222  172201              ARET5:  ADDF    AC1,AC2         ;ADD AC1 BY AC2
000933  005224  170237  002362              STFPS   FPS             ;STORE FLOATING POINT STATUS
000934  005230  023737  002362  002400      CMP     FPS,$FPS        ;CHECK FPS
```



```
000935  005236  001403                      BEQ     AERR5           ;BRANCH IF OK
000936  005240  174237  002406              STF     AC2,ANS1        ;SAVE FPU ANSWER
000937  005244  104001                      ERROR   1               ;FPS ERROR
000938
000939  005246  005737  002400      AERR5:  TST     $FPS            ;ERROR BIT SET ?
000940  005252  100014                      BPL     ATST5           ;NO, DONT GET FEC/FEA
000941  005254  170337  002364              STST    FEC             ;YES, CHECK STATUS
000942
000943  005260  023737  002364  002402      CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
000944  005266  001401                      BEQ     1$              ;BRANCH IF OK
000945  005270  104023                      ERROR   23              ;FEC IS WRONG
000946
000947  005272  023737  002366  002404  1$: CMP     FEA,$FEA        ;CHECK FLOATING PC
000948  005300  001401                      BEQ     ATST5           ;BRANCH IF OK
000949  005302  104023                      ERROR   23              ;WRONG ADDRESS IN FEA
000950
000951  005304  173702              ATST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
000952  005306  170000                      CFCC                    ;COPY FLOATING CONDITION CODES
000953  005310  001427                      BEQ     AEND5           ;ANSWERS CHECK
000954                                      ;COMPENSATE FOR FORTRAN INACCURACIES.
000955  005312  174237  002406              STF     AC2,ANS1        ;SAVE FPU ANSWER
000956  005316  062737  000001  002410      ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
000957  005324  005537  002406              ADC     ANS1
000958  005330  173737  002406              CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
000959  005334  170000                      CFCC                    ;COPY FLOATING CONDITION CODES
000960  005336  001414                      BEQ     AEND5           ;BRANCH IF OK
000961  005340  162737  000002  002410      SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
000962  005346  005637  002406              SBC     ANS1
000963  005352  173737  002406              CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
000964  005356  170003                      CFCC                    ;COPY FLOATING CONDITION CODES
000965  005360  001403                      BEQ     AEND5           ;BRANCH IF OK
000966  005362  174237  002406              STF     AC2,ANS1        ;SAVE FPU ANSWER
000967  005366  104005                      EPROR   5               ;FPU AND FORTRAN DISAGREE
000968
000969  005370  005037  002362      AEND5:  CLR     FPS             ;CLR FPU FPS BUFFER
000970
000971
000972
000973
000974
000975                                      ;;*****************************************************************
000976                                      ;*TEST 6         EXERCISE ADDD, ALL INTERRUPTS ON, TRUNCATE MODE
000977                                      ;;*****************************************************************
000978  005374  000004              TST6:   SCOPE
000979  005376  012737  005530  002376      MOV     #ARET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
000980  005404  012737  007640  002400      MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
000981  005412  005037  002402              CLR     $FEC            ;CLR FORTRAN FEC
000982  005416  005037  002404              CLR     $FEA            ;CLR FORTRAN FEA
000983  005422  005037  002362              CLR     FPS             ;CLR FPU FPS BUFFER
000984  005426  005037  002364              CLR     FEC             ;CLR FPU FEC BUFFER
000985  005432  005037  002366              CLR     FEA             ;CLR FPU FEA BUFFER
000986  005436  004737  023332              JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
000987  005442  002426  002436              .WORD   LONUM,HINUM     ;
000988  005446  004437  023506              JSR     R4,$POLSH       ;ENTER POLISH MODE
000989  005452  023510  002426              $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
000990  005456  023510  002436              $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
```

```
  991  005462  023566                        $ADD                    ;ADDRESS OF FORTRAN ADD
  992  005464  073540  002416                $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
  993
  994  005470  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
  995  005474  170127  040200                LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
  996  005500  172437  002426                LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
  997  005504  172537  002436                LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
  998  005510  172737  002416                LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
  999  005514  170127  007640                LDFPS   #007640         ;TURN INTERRUPTS ON
 1000  005520  012737  005526  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
 1001
 1002                                        ;*************************************************************
 1003
 1004  005526  172600                        LDD     AC0,AC2         ;LOAD AC0 INTO AC2
 1005  005530  172201                ARET6:  ADDD    AC1,AC2         ;ADD AC1 BY AC2
 1006  005532  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
 1007  005536  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
 1008  005544  001403                        BEQ     AERR6           ;BRANCH IF OK
 1009  005546  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
 1010  005552  104017                        ERROR   17              ;FPS ERROR
 1011
 1012  005554  005737  002400        AERR6:  TST     $FPS            ;ERROR BIT SET ?
 1013  005560  100014                        BPL     ATST6           ;NO, DONT GET FEC/FEA
 1014  005562  173337  002364                STST    FEC             ;YES, CHECK STATUS
 1015
 1016  005566  023737  002364  002402        CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
 1017  005574  001401                        BEQ     1S              ;BRANCH IF OK
 1018  005576  104027                        ERROR   27              ;FEC IS WRONG
 1019
 1020  005600  023737  002366  002404  1S:   CMP     FEA,$FEA        ;CHECK FLOATING PC
 1021  005606  001401                        BEQ     ATST6           ;BRANCH IF OK
 1022  005610  104027                        ERROR   27              ;WRONG ADDRESS IN FEA
 1023
 1024  005612  173702                ATST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
 1025  005614  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
 1026  005616  001437                        BEQ     AEND6           ;ANSWERS CHECK
 1027                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
 1028  005620  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
 1029  005624  062737  000001  002414        ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
 1030  005632  005537  002412                ADC     ANS1+4
 1031  005636  005537  002410                ADC     ANS1+2
 1032  005642  005537  002406                ADC     ANS1
 1033  005646  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1034  005652  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
 1035  005654  001420                        BEQ     AEND6           ;BRANCH IF OK
 1036  005656  162737  000002  002414        SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 1037  005664  005637  002412                SBC     ANS1+4
 1038  005670  005637  002410                SBC     ANS1+2
 1039  005674  005637  002406                SBC     ANS1
 1040  005700  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1041  005704  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
 1042  005706  001403                        BEQ     AEND6           ;BRANCH IF OK
 1043  005710  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
 1044  005714  104006                        ERROR   6               ;FPU AND FORTRAN DISAGREE
 1045
 1046  005716  005037  002362        AEND6:  CLR     FPS             ;CLR FPU FPS BUFFER
```



```
 1047
 1048
 1049
 1050
 1051
 1052                                        ;;*************************************************************
 1053                                        ;*TEST 7        EXERCISE ADDF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
 1054                                        ;;*************************************************************
 1055  005722  000004                TST7:   SCOPE
 1056  005724  012737  006056  002376        MOV     #ARET7,EXPFEA   ;ADDR OF INSTR BEING TESTED
 1057  005732  012737  004440  002400        MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
 1058  005740  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
 1059  005744  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
 1060  005750  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
 1061  005754  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
 1062  005760  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
 1063  005764  004737  023342                JSR     PC,PANDL2       ;GET RANDOM INPUT DATA
 1064  005770  002426  002436                ,WORD   LONUM,HINUM     ;
 1065  005774  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
 1066  006000  023510  002426                $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
 1067  006004  023510  002436                $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
 1068  006010  023566                        $ADD                    ;ADDRESS OF FORTRAN ADD
 1069  006012  073540  002416                $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
 1070
 1071  006016  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
 1072  006022  170127  040000                LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
 1073  006026  172437  002426                LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
 1074  006032  172537  002436                LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
 1075  006036  172737  002416                LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
 1076  006042  170127  004440                LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
 1077  006046  012737  006054  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
 1078
 1079                                        ;*************************************************************
 1080
 1081  006054  172600                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
 1082  006056  172201                ARET7:  ADDF    AC1,AC2         ;ADD AC1 BY AC2
 1083  006060  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
 1084  006064  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
 1085  006072  001403                        BEQ     ATST7           ;BRANCH IF OK
 1086  006074  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
 1087  006100  104001                        ERROR   1               ;FPS ERROR
 1088
 1089  006102  173702                ATST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
 1090  006104  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
 1091  006106  001427                        BEQ     AEND7           ;ANSWERS CHECK
 1092                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
 1093  006110  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
 1094  006114  062737  000001  002410        ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
 1095  006122  005537  002406                ADC     ANS1
 1096  006126  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1097  006132  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
 1098  006134  001414                        BEQ     AEND7           ;BRANCH IF OK
 1099  006136  162737  000002  002410        SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 1100  006144  005637  002406                SBC     ANS1
 1101  006150  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1102  006154  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
```

```
1103  006156  001403                  BEQ     AEND7           ;BRANCH IF OK
1104  006160  174237  002406          STF     AC2,ANS1        ;SAVE FPU ANSWER
1105  006164  104005                  ERROR   5               ;FPU AND FORTRAN DISAGREE
1106
1107  006166  005037  002362  AEND7:  CLR     FPS             ;CLR FPU FPS BUFFER
1108
1109
1110
1111
1112                                  ;;*************************************************************
1113                                  ;*TEST 10        EXERCISE ADDD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
1114                                  ;;*************************************************************
1115  006172  000004          TST10:  SCOPE
1116  006174  012737  006326  002376          MOV     #ARET10,EXPFEA  ;ADDR OF INSTR BEING TESTED
1117  006202  012737  004640  002400          MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1118  006210  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
1119  006214  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
1120  006220  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
1121  006224  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
1122  006230  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
1123  006234  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1124  006240  002426  002436                  .WORD   LONUM,HINUM     ;
1125  006244  004437  023506                  JSP     R4,$POLSH       ;ENTER POLISH MODE
1126  006250  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1127  006254  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1128  006260  023566                          $ADD                    ;ADDRESS OF FORTRAN ADD
1129  006262  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1130
1131  006266  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1132  006272  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
1133  006276  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1134  006302  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1135  006306  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1136  006312  170127  004640                  LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
1137  006316  012737  006324  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1138
1139                                  ;*************************************************************
1140
1141  006324  172600                  ARET10: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1142  006326  172201                          ADDD    AC1,AC2         ;ADD AC1 BY AC2
1143  006330  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1144  006334  023737  002362  002400          CMP     FPS,#FPS        ;CHECK FPS
1145  006342  001403                          BEQ     ATST10          ;BRANCH IF OK
1146  006344  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1147  006350  104017                          ERROR   17              ;FPS ERROR
1148
1149  006352  173702                  ATST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1150  006354  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1151  006356  001437                          BEQ     AEND10          ;ANSWERS CHECK
1152                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1153  006360  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1154  006364  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
1155  006372  005537  002412                  ADC     ANS1+4
1156  006376  005537  002410                  ADC     ANS1+2
1157  006402  005537  002406                  ADC     ANS1
1158  006406  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
```



```
1159  006412  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
1160  006414  001422                  BEQ     AEND10          ;BRANCH IF OK
1161  006416  162737  000002  002414  SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
1162  006424  005637  002412                  SBC     ANS1+4
1163  006430  005637  002410                  SBC     ANS1+2
1164  006434  005637  002406                  SBC     ANS1
1165  006440  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
1166  006444  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1167  006446  001403                          BEQ     AEND10          ;BRANCH IF OK
1168  006450  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1169  006454  104006                          ERROR   6               ;FPU AND FORTRAN DISAGREE
1170
1171  006456  005037  002362  AEND10: CLR     FPS             ;CLR FPU FPS BUFFER
1172
```

```
1173                                    ;;*****************************************************************
1174                                    ;*TEST 11        EXERCISE SUBF, ALL INTERRUPTS ON, ROUNDING MODE
1175                                    ;;*****************************************************************
1176  006462  000004                    TST11:  SCOPE
1177  006464  012737  006616  002376            MOV     #ARET11,EXPFEA  ;ADDR OF INSTR BEING TESTED
1178  006472  012737  007400  002400            MOV     #007400,@FPS    ;SET IE BITS IN FORTRAN ANSWER
1179  006500  005037  002402                    CLR     @FEC            ;CLR FORTRAN FEC
1180  006504  005037  002404                    CLR     @FEA            ;CLR FORTRAN FEA
1181  006510  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
1182  006514  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
1183  006520  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
1184  006524  004737  023342                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1185  006530  002426  002436                    .WORD   LONUM,HINUM     ;
1186  006534  004437  023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE
1187  006540  023510  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1188  006544  023510  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1189  006550  023562                            $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1190  006552  023540  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1191
1192  006556  013700  002400                    MOV     @FPS,R0         ;DISPLAY FLOATING POINT STATUS
1193  006562  170127  040000                    LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
1194  006566  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1195  006572  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1196  006576  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1197  006602  170127  007400                    LDFPS   #007400         ;TURN INTERRUPTS ON
1198  006606  012737  006614  001110            MOV     #.+6,@LPADR     ;RESET LOOP ADDRESS
1199
1200                                    ;*****************************************************************
1201
1202  006614  172600                    ARET11: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1203  006616  173201                            SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
1204  006620  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1205  006624  023737  002362  002400            CMP     FPS,@FPS        ;CHECK FPS
1206  006632  001403                            BEQ     AERR11          ;BRANCH IF OK
1207  006634  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
1208  006640  104002                            ERROR   2               ;FPS ERROR
1209
1210  006642  005737  002400            AERR11: TST     @FPS            ;ERROR BIT SET ?
1211  006646  100014                            BPL     ATST11          ;NO, DONT GET FEC/FEA
1212  006650  170337  002364                    STST    FEC             ;YES, CHECK STATUS
1213
1214  006654  023737  002364  002402            CMP     FEC,@FEC        ;CHECK THE FLOATING EXCEPTION CODES
1215  006662  001401                            BEQ     1$              ;BRANCH IF OK
1216  006664  104024                            ERROR   24              ;FEC IS WRONG
1217
1218  006666  023737  002366  002404  1$:       CMP     FEA,@FEA        ;CHECK FLOATING PC
1219  006674  001401                            BEQ     ATST11          ;BRANCH IF OK
1220  006676  104024                            ERROR   24              ;WRONG ADDRESS IN FEA
1221
1222  006700  173702                    ATST11: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1223  006702  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
1224  006704  001416                            BEQ     AEND11          ;ANSWERS CHECK
1225                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
1226  006706  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
1227  006712  162737  000001  002410            SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1228  006720  005637  002406                    SBC     ANS1
```



```
1229  006721  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1230  006730  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
1231  006732  001403                            BEQ     AEND11          ;BRANCH IF OK
1232  006734  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
1233  006740  104007                            ERROR   7               ;FPU AND FORTRAN DISAGREE
1234
1235  006742  005037  002362            AEND11: CLR     FPS             ;CLR FPU FPS BUFFER
1236
1237
1238
1239
1240                                    ;;*****************************************************************
1241                                    ;*TEST 12        EXERCISE SUBD, ALL INTERRUPTS ON, ROUNDING MODE
1242                                    ;;*****************************************************************
1243  006746  000004                    TST12:  SCOPE
1244  006750  012737  007102  002376            MOV     #ARET12,EXPFEA  ;ADDR OF INSTR BEING TESTED
1245  006756  012737  007600  002400            MOV     #007600,@FPS    ;SET IE BITS IN FORTRAN ANSWER
1246  006764  005037  002402                    CLR     @FEC            ;CLR FORTRAN FEC
1247  006770  005037  002404                    CLR     @FEA            ;CLR FORTRAN FEA
1248  006774  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
1249  007000  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
1250  007004  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
1251  007010  004737  023332                    JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1252  007014  002426  002436                    .WORD   LONUM,HINUM     ;
1253  007020  004437  023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE
1254  007024  023510  002426                    $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1255  007030  023510  002436                    $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1256  007034  023562                            $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1257  007036  023540  002416                    $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1258
1259  007042  013700  002400                    MOV     @FPS,R0         ;DISPLAY FLOATING POINT STATUS
1260  007046  170127  040200                    LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
1261  007052  172437  002426                    LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1262  007056  172537  002436                    LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1263  007062  172737  002416                    LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1264  007066  170127  007600                    LDFPS   #007600         ;TURN INTERRUPTS ON
1265  007072  012737  007100  001110            MOV     #.+6,@LPADR     ;RESET LOOP ADDRESS
1266
1267                                    ;*****************************************************************
1268
1269  007100  172600                    ARET12: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1270  007102  173201                            SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
1271  007104  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1272  007110  023737  002362  002400            CMP     FPS,@FPS        ;CHECK FPS
1273  007116  001403                            BEQ     AERR12          ;BRANCH IF OK
1274  007120  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
1275  007124  104020                            ERROR   20              ;FPS ERROR
1276
1277  007126  005737  002400            AERR12: TST     @FPS            ;ERROR BIT SET ?
1278  007132  100014                            BPL     ATST12          ;NO, DONT GET FEC/FEA
1279  007134  170337  002364                    STST    FEC             ;YES, CHECK STATUS
1280
1281  007140  023737  002364  002402            CMP     FEC,@FEC        ;CHECK THE FLOATING EXCEPTION CODES
1282  007146  001401                            BEQ     1$              ;BRANCH IF OK
1283  007150  104030                            ERROR   30              ;FEC IS WRONG
1284
```

```
1285  007152  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
1286  007160  001401                          BEQ     ATST12          ;BRANCH IF OK
1287  007162  104030                          ERROR   30              ;WRONG ADDRESS IN FEA
1288
1289  007164  173702                  ATST12: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1290  007166  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1291  007170  001422                          BEQ     AEND12          ;ANSWERS CHECK
1292                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1293  007172  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1294  007176  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
1295  007204  005637  002412                  SBC     ANS1+4
1296  007210  005637  002410                  SBC     ANS1+2
1297  007214  005637  002406                  SBC     ANS1
1298  007220  173737  002406                  CMPD    ANS1,AC1        ;CHECK ANSWERS AGAIN
1299  007224  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1300  007226  001403                          BEQ     AEND12          ;BRANCH IF OK
1301  007230  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1302  007234  104010                          ERROR   10              ;FPU AND FORTRAN DISAGREE
1303
1304  007236  005037  002362          AEND12: CLR     FPS             ;CLR FPU FPS BUFFER
1305
1306
1307
1308
1309                                          ;;*************************************************************
1310                                          ;*TEST 13        EXERCISE SUBF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
1311                                          ;;*************************************************************
1312  007242  000004                  TST13:  SCOPE
1313  007244  012737  007376  002376          MOV     #ARET13,EXPFEA  ;ADDR OF INSTR BEING TESTED
1314  007252  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1315  007260  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
1316  007264  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
1317  007270  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
1318  007274  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
1319  007300  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
1320  007304  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1321  007310  002426  002436                  .WORD   LONUM,HINUM     ;
1322  007314  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
1323  007320  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1324  007324  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1325  007330  023562                          $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1326  007332  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1327
1328  007336  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1329  007342  170127  040000                  LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
1330  007346  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1331  007352  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1332  007356  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1333  007362  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
1334  007366  012737  007374  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1335
1336                                          ;*************************************************************
1337
1338  007374  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1339  007376  173201                  ARET13: SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
1340  007400  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
```



```
1341  007404  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
1342  007412  001403                          BEQ     ATST13          ;BRANCH IF OK
1343  007414  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1344  007420  104002                          ERROR   2               ;FPS ERROR
1345
1346  007422  173702                  ATST13: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1347  007424  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1348  007426  001416                          BEQ     AEND13          ;ANSWERS CHECK
1349                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1350  007430  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1351  007434  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1352  007442  005637  002406                  SBC     ANS1
1353  007446  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1354  007452  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1355  007454  001403                          BEQ     AEND13          ;BRANCH IF OK
1356  007456  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1357  007462  104007                          ERROR   7               ;FPU AND FORTRAN DISAGREE
1358
1359  007464  005037  002362          AEND13: CLR     FPS             ;CLR FPU FPS BUFFER
1360
1361
1362
1363                                          ;;*************************************************************
1364                                          ;*TEST 14        EXERCISE SUBD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
1365                                          ;;*************************************************************
1366
1367  007470  000004                  TST14:  SCOPE
1368  007472  012737  007624  002376          MOV     #ARET14,EXPFEA  ;ADDR OF INSTR BEING TESTED
1369  007500  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1370  007506  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
1371  007512  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
1372  007516  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
1373  007522  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
1374  007526  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
1375  007532  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1376  007536  002426  002436                  .WORD   LONUM,HINUM     ;
1377  007542  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
1378  007546  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1379  007552  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1380  007556  023562                          $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1381  007560  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1382
1383  007564  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1384  007570  170127  042000                  LDFPS   #042000         ;LOAD FPS, INTERRUPT DISABLE AND FD
1385  007574  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1386  007600  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1387  007604  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1388  007610  170127  004600                  LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
1389  007614  012737  007622  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1390
1391                                          ;*************************************************************
1392
1393  007622  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1394  007624  173201                  ARET14: SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
1395  007626  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1396  007632  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
```

```
1397  007640  001403                        BEQ     ATST14          ;BRANCH IF OK
1398  007642  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1399  007646  104020                        ERROR   20              ;FPS ERROR
1400
1401  007650  173702                ATST14: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1402  007652  170003                        CFCC                    ;COPY FLOATING CONDITION CODES
1403  007654  001422                        BEQ     AEND14          ;ANSWERS CHECK
1404                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
1405  007656  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1406  007662  162737  000001  002414        SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
1407  007670  005637  002412                SBC     ANS1+4
1408  007674  005637  002410                SBC     ANS1+2
1409  007700  005637  002406                SBC     ANS1
1410  007704  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
1411  007710  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1412  007712  001403                        BEQ     AEND14          ;BRANCH IF OK
1413  007714  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1414  007720  104010                        ERROR   10              ;FPU AND FORTRAN DISAGREE
1415
1416  007722  005037  002362        AEND14: CLR     FPS             ;CLR FPU FPS BUFFER
1417
1418
1419
1420
1421                                        ;;*************************************************************
1422                                        ;*TEST 15        EXERCISE SUBF, ALL INTERRUPTS ON, TRUNCATE MODE
1423                                        ;;*************************************************************
1424  007726  000004                TST15:  SCOPE
1425  007730  012737  010062  002376        MOV     #ARET15,EXPFEA  ;ADDR OF INSTR BEING TESTED
1426  007736  012737  007440  002400        MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1427  007744  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
1428  007750  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
1429  007754  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
1430  007760  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
1431  007764  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
1432  007770  004737  023342                JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1433  007774  002426  002436                .WORD   LONUM,HINUM     ;
1434  010000  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
1435  010004  023510  002426                $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1436  010010  023510  002436                $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1437  010014  023562                        $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1438  010016  023540  002416                $POPA   ,ANS2           ;POP 2 WORDS AND EXIT-POLISH MODE
1439
1440  010022  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1441  010026  170127  040000                LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
1442  010032  172437  002426                LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1443  010036  172537  002436                LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1444  010042  172737  002416                LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1445  010046  170127  007440                LDFPS   #007440         ;TURN INTERRUPTS ON
1446  010052  012737  010060  001110        MOV     #,+6,$LPADR     ;RESET LOOP ADDRESS
1447
1448                                        ;*************************************************
1449
1450  010060  172600                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1451  010062  173201                ARET15: SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
1452  010064  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
```



```
1453  010070  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
1454  010076  001403                        BEQ     AERR15          ;BRANCH IF OK
1455  010100  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
1456  010104  104002                        ERROR   2               ;FPS ERROR
1457
1458  010106  005737  002400        AERR15: TST     $FPS            ;ERROR BIT SET ?
1459  010112  100014                        BPL     ATST15          ;NO, DONT GET FEC/FEA
1460  010114  170337  002364                STST    FEC             ;YES, CHECK STATUS
1461
1462  010120  023737  002364  002402        CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1463  010126  001401                        BEQ     15              ;BRANCH IF OK
1464  010130  104024                        ERROR   24              ;FEC IS WRONG
1465
1466  010132  023737  002366  002404  15:    CMP     FEA,$FEA        ;CHECK FLOATING PC
1467  010140  001403                        BEQ     ATST15          ;BRANCH IF OK
1468  010142  104024                        ERROR   24              ;WRONG ADDRESS IN FEA
1469
1470  010144  173702                ATST15: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1471  010146  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1472  010150  001427                        BEQ     AEND15          ;ANSWERS CHECK
1473                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
1474  010152  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
1475  010156  062737  000001  002410        ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
1476  010164  005537  002406                ADC     ANS1
1477  010170  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1478  010174  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1479  010176  001414                        BEQ     AEND15          ;BRANCH IF OK
1480  010200  162737  000002  002410        SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
1481  010206  005637  002406                SBC     ANS1
1482  010212  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1483  010216  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1484  010220  001403                        BEQ     AEND15          ;BRANCH IF OK
1485  010222  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
1486  010226  104007                        ERROR   7               ;FPU AND FORTRAN DISAGREE
1487
1488  010230  005037  002362        AEND15: CLR     FPS             ;CLR FPU FPS BUFFER
1489
1490
1491
1492
1493                                        ;;*************************************************************
1494                                        ;*TEST 16        EXERCISE SUBD, ALL INTERRUPTS ON, TRUNCATE MODE
1495                                        ;;*************************************************************
1496  010234  000004                TST16:  SCOPE
1497  010236  012737  010370  002376        MOV     #ARET16,EXPFEA  ;ADDR OF INSTR BEING TESTED
1498  010244  012737  007640  002400        MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1499  010252  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
1500  010256  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
1501  010262  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
1502  010266  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
1503  010272  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
1504  010276  004737  023332                JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1505  010302  002426  002436                .WORD   LONUM,HINUM     ;
1506  010306  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
1507  010312  023510  002426                $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1508  010316  023510  002436                $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
```

```
1509  010322  023562                        $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1510  010324  023540  002416                $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1511
1512  010330  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1513  010334  170127  040200                LDFPS   #40200          ;LOAD FPS, INTERRUPT DISABLE AND FD
1514  010340  172437  002426                LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1515  010344  172537  002436                LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1516  010350  172737  002416                LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1517  010354  170127  007640                LDFPS   #7640           ;TURN INTERRUPTS ON
1518  010360  012737  010366  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1519
1520                                        ;*************************************************************
1521
1522  010366  172600                        LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1523  010370  173201                ARET16: SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
1524  010372  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
1525  010376  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
1526  010404  001403                        BEQ     AERR16          ;BRANCH IF OK
1527  010406  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1528  010412  104020                        ERROR   20              ;FPS ERROR
1529
1530  010414  005737  002400        AERR16: TST     $FPS            ;ERROR BIT SET ?
1531  010420  100014                        BPL     ATST16          ;NO, DONT GET FEC/FEA
1532  010422  170337  002364                STST    FEC             ;YES, CHECK STATUS
1533
1534  010426  023737  002364  002402        CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1535  010434  001401                        BEQ     1$              ;BRANCH IF OK
1536  010436  104030                        ERROR   30              ;FEC IS WRONG
1537
1538  010440  023737  002366  002404  1$:    CMP     FEA,$FEA        ;CHECK FLOATING PC
1539  010446  001401                        BEQ     ATST16          ;BRANCH IF OK
1540  010450  104030                        ERROR   30              ;WRONG ADDRESS IN FEA
1541
1542  010452  173702                ATST16: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1543  010454  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1544  010456  001437                        BEQ     AEND16          ;ANSWERS CHECK
1545                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
1546  010460  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1547  010464  062737  000001  002414        ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
1548  010472  005537  002412                ADC     ANS1+4
1549  010476  005537  002410                ADC     ANS1+2
1550  010502  005537  002406                ADC     ANS1
1551  010506  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
1552  010512  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1553  010514  001424                        BEQ     AEND16          ;BRANCH IF OK
1554  010516  162737  000002  002414        SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
1555  010524  005637  002412                SBC     ANS1+4
1556  010530  005637  002410                SBC     ANS1+2
1557  010534  005637  002406                SBC     ANS1
1558  010540  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
1559  010544  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1560  010546  001403                        BEQ     AEND16          ;BRANCH IF OK
1561  010550  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
1562  010554  104010                        ERROR   10              ;FPU AND FORTRAN DISAGREE
1563
1564  010556  005037  002362        AEND16: CLR     FPS             ;CLR FPU FPS BUFFER
```



```
1565
1566
1567
1568
1569                                        ;;*************************************************************
1570                                        ;*TEST 17        EXERCISE SUBF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
1571                                        ;;*************************************************************
1572  010562  000004                TST17:  SCOPE
1573  010564  012737  010716  002376        MOV     #ARET17,EXPFEA  ;ADDR OF INSTR BEING TESTED
1574  010572  012737  004440  002400        MOV     #4440,$FPS      ;SET TE BITS IN FORTRAN ANSWER
1575  010600  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
1576  010604  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
1577  010610  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
1578  010614  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
1579  010620  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
1580  010624  004737  023342                JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1581  010630  002426  002436                .WORD   LONUM,HINUM     ;
1582  010634  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
1583  010640  023510  002426                $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1584  010644  023510  002436                $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1585  010650  023562                        $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1586  010652  023540  002416                $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1587
1588  010656  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1589  010662  170127  040000                LDFPS   #40000          ;LOAD FPS, INTERRUPT DISABLE
1590  010666  172437  002426                LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1591  010672  172537  002436                LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1592  010676  172737  002416                LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1593  010702  170127  004440                LDFPS   #4440           ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
1594  010706  012737  010714  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1595
1596                                        ;*************************************************************
1597
1598  010714  172600                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1599  010716  173201                ARET17: SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
1600  010720  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
1601  010724  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
1602  010732  001403                        BEQ     ATST17          ;BRANCH IF OK
1603  010734  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
1604  010740  104002                        ERROR   2               ;FPS ERROR
1605
1606  010742  173702                ATST17: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1607  010744  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1608  010746  001427                        BEQ     AEND17          ;ANSWERS CHECK
1609                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
1610  010750  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
1611  010754  062737  000001  002410        ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
1612  010762  005537  002406                ADC     ANS1
1613  010766  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1614  010772  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1615  010774  001414                        BEQ     AEND17          ;BRANCH IF OK
1616  010776  162737  000002  002410        SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
1617  011004  005637  002406                SBC     ANS1
1618  011010  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1619  011014  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
1620  011016  001403                        BEQ     AEND17          ;BRANCH IF OK
```

```
 1621  011020  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 1622  011024  104007                          ERROR   7               ;FPU AND FORTRAN DISAGREE
 1623
 1624  011026  005037  002362          AEND17: CLR     FPS             ;CLR FPU FPS BUFFER
 1625
 1626
 1627
 1628                                    ;;*************************************************************
 1629                                    ;*TEST 20        EXERCISE SUBD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
 1630                                    ;;*************************************************************
 1631
 1632  011032  000004                    TST20:  SCOPE
 1633  011034  012737  011166  002376            MOV     #ARET20,EXPFEA  ;ADDR OF INSTR BEING TESTED
 1634  011042  012737  004640  002400            MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
 1635  011050  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
 1636  011054  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
 1637  011060  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
 1638  011064  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
 1639  011070  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
 1640  011074  004737  023332                    JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
 1641  011100  002426  002436                    .WORD   LONUM,HINUM     ;
 1642  011104  004437  023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE
 1643  011110  023510  002426                    $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
 1644  011114  023510  002436                    $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
 1645  011120  023562                            $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
 1646  011122  023540  002416                    $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
 1647
 1648  011126  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
 1649  011132  170127  040200                    LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
 1650  011136  172437  002426                    LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
 1651  011142  172537  002436                    LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
 1652  011146  172737  002416                    LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
 1653  011152  170127  004640                    LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
 1654  011156  012737  011164  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
 1655                                    ;*************************************************************
 1656
 1657
 1658  011164  172600                    ARET20: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
 1659  011166  173201                            SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
 1660  011170  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
 1661  011174  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
 1662  011202  001403                            BEQ     ATST20          ;BRANCH IF OK
 1663  011204  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
 1664  011210  104020                            ERROR   20              ;FPS ERROR
 1665
 1666  011212  173702                    ATST20: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
 1667  011214  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 1668  011216  001437                            BEQ     AEND20          ;ANSWERS CHECK
 1669                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
 1670  011220  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
 1671  011224  062737  000001  002414            ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
 1672  011232  005537  002412                    ADC     ANS1+4
 1673  011236  005537  002410                    ADC     ANS1+2
 1674  011242  005537  002406                    ADC     ANS1
 1675  011246  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1676  011252  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
```



```
 1677  011254  001420                            BEQ     AEND20          ;BRANCH IF OK
 1678  011256  162737  000002  002414            SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 1679  011264  005637  002412                    SBC     ANS1+4
 1680  011270  005637  002410                    SBC     ANS1+2
 1681  011274  005637  002406                    SBC     ANS1
 1682  011300  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1683  011304  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 1684  011306  001403                            BEQ     AEND20          ;BRANCH IF OK
 1685  011310  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
 1686  011314  104010                            EPROP   10              ;FPU AND FORTRAN DISAGREE
 1687
 1688  011316  005037  002362          AEND20: CLR     FPS             ;CLR FPU FPS BUFFER
 1689
```

```
1690                                    ;;*****************************************************************
1691                                    ;*TEST 21        EXERCISE MULF, ALL INTERRUPTS ON, ROUNDING MODE
1692                                    ;;*****************************************************************
1693  011322  000004                   TST21:  SCOPE
1694  011324  012737  011456  002376            MOV     #MRET1,EXPFEA   ;ADDR OF INSTR BEING TESTED
1695  011332  012737  007400  002400            MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1696  011340  005037  002402                   CLR     $FEC            ;CLR FORTRAN FEC
1697  011344  005037  002404                   CLR     $FEA            ;CLR FORTRAN FEA
1698  011350  005037  002362                   CLR     FPS             ;CLR FPU FPS BUFFER
1699  011354  005037  002364                   CLR     FEC             ;CLR FPU FEC BUFFER
1700  011360  005037  002366                   CLR     FEA             ;CLR FPU FEA BUFFER
1701  011364  004737  023342                   JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1702  011370  002426  002436                   .WORD   LONUM,HINUM     ;
1703  011374  004437  023506                   JSR     R4,$POLSH       ;ENTER POLISH MODE
1704  011400  023510  002426                   $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1705  011404  023510  002436                   $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1706  011410  025244                           $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
1707  011412  023540  002416                   $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1708
1709  011416  013700  002400                   MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1710  011422  170127  040000                   LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
1711  011426  172437  002426                   LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1712  011432  172537  002436                   LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1713  011436  172737  002416                   LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1714  011442  170127  007400                   LDFPS   #007400         ;TURN INTERRUPTS ON
1715  011446  012737  011454  001110           MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1716
1717                                    ;*****************************************************************
1718
1719  011454  172600                   MRET1:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1720  011456  171201                           MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
1721  011460  170237  002362                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1722  011464  023737  002362  002400           CMP     FPS,$FPS        ;CHECK FPS
1723  011472  001403                           BEQ     MERR1           ;BRANCH IF OK
1724  011474  174237  002406                   STF     AC2,ANS1        ;SAVE FPU ANSWER
1725  011500  104003                           ERROR   3               ;FPS ERROR
1726
1727  011502  005737  002400           MERR1:  TST     $FPS            ;ERROR BIT SET ?
1728  011506  100014                           BPL     MTST1           ;NO, DONT GET FEC/FEA
1729  011510  170337  002364                   STST    FEC             ;YES, CHECK STATUS
1730
1731  011514  023737  002364  002402           CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1732  011522  001401                           BEQ     1$              ;BRANCH IF OK
1733  011524  104025                           ERROR   25              ;FEC IS WRONG
1734
1735  011526  023737  002366  002404   1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
1736  011534  001401                           BEQ     MTST1           ;BRANCH IF OK
1737  011536  104025                           ERROR   25              ;WRONG ADDRESS IN FEA
1738
1739  011540  173702                   MTST1:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1740  011542  170000                           CFCC                    ;COPY FLOATING CONDITION CODES
1741  011544  001416                           BEQ     MEND1           ;ANSWERS CHECK
1742                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1743  011546  174237  002406                   STF     AC2,ANS1        ;SAVE FPU ANSWER
1744  011552  162737  000001  002410           SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1745  011560  005637  002406                   SBC     ANS1
```



```
1746  011564  173737  002406                   CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1747  011570  170000                           CFCC                    ;COPY FLOATING CONDITION CODES
1748  011572  001403                           BEQ     MEND1           ;BRANCH IF OK
1749  011574  174237  002406                   STF     AC2,ANS1        ;SAVE FPU ANSWER
1750  011600  104011                           ERROR   11              ;FPU AND FORTRAN DISAGREE
1751
1752  011602  005037  002362           MEND1:  CLR     FPS             ;CLEAR FPP FPS BUFFER
1753
1754
1755
1756                                    ;;*****************************************************************
1757                                    ;*TEST 22        EXERCISE MULD, ALL INTERRUPTS ON, ROUNDING MODE
1758                                    ;;*****************************************************************
1759
1760  011606  000004                   TST22:  SCOPE
1761  011610  012737  011742  002376            MOV     #MRET2,EXPFEA   ;ADDR OF INSTR BEING TESTED
1762  011616  012737  007600  002400            MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1763  011624  005037  002402                   CLR     $FEC            ;CLR FORTRAN FEC
1764  011630  005037  002404                   CLR     $FEA            ;CLR FORTRAN FEA
1765  011634  005037  002362                   CLR     FPS             ;CLR FPU FPS BUFFER
1766  011640  005037  002364                   CLR     FEC             ;CLR FPU FEC BUFFER
1767  011644  005037  002366                   CLR     FEA             ;CLR FPU FEA BUFFER
1768  011650  004737  023332                   JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1769  011654  002426  002436                   .WORD   LONUM,HINUM     ;
1770  011660  004437  023506                   JSR     R4,$POLSH       ;ENTER POLISH MODE
1771  011664  023510  002426                   $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1772  011670  023510  002436                   $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1773  011674  025244                           $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
1774  011676  023540  002416                   $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1775
1776  011702  013700  002400                   MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1777  011706  170127  040200                   LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
1778  011712  172437  002426                   LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1779  011716  172537  002436                   LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1780  011722  172737  002416                   LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1781  011726  170127  007600                   LDFPS   #007600         ;TURN INTERRUPTS ON
1782  011732  012737  011740  001110           MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1783
1784                                    ;*****************************************************************
1785
1786  011740  172600                   MRET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1787  011742  171201                           MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
1788  011744  170237  002362                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1789  011750  023737  002362  002400           CMP     FPS,$FPS        ;CHECK FPS
1790  011756  001403                           BEQ     MERR2           ;BRANCH IF OK
1791  011760  174237  002406                   STD     AC2,ANS1        ;SAVE FPU ANSWER
1792  011764  104021                           ERROR   21              ;FPS ERROR
1793
1794  011766  005737  002400           MERR2:  TST     $FPS            ;ERROR BIT SET ?
1795  011772  100014                           BPL     MTST2           ;NO, DONT GET FEC/FEA
1796  011774  170337  002364                   STST    FEC             ;YES, CHECK STATUS
1797
1798  012000  023737  002364  002402           CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1799  012006  001401                           BEQ     1$              ;BRANCH IF OK
1800  012010  104031                           ERROR   31              ;FEC IS WRONG
1801                                                                    .
```

```
1802  012012  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
1803  012020  001401                          BEQ     MTST2           ;BRANCH IF OK
1804  012022  104031                          ERROR   31              ;WRONG ADDRESS IN FEA
1805
1806  012024  173702                  MTST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1807  012026  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1808  012030  001422                          BEQ     MEND2           ;ANSWERS CHECK
1809                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1810  012032  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1811  012036  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
1812  012044  005637  002412                  SBC     ANS1+4
1813  012050  005637  002410                  SBC     ANS1+2
1814  012054  005637  002406                  SBC     ANS1
1815  012060  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
1816  012064  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
1817  012066  001403                          BEQ     MEND2           ;BRANCH IF OK
1818  012070  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
1819  012074  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
1820
1821  012076  005037  002362          MEND2:  CLR     FPS             ;CLEAR FPP FPS BUFFER
1822
1823
1824
1825
1826                                          ;;*************************************************************
1827                                  ;*TEST 23        EXERCISE MULF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
1828                                          ;;*************************************************************
1829  012102  000004                  TST23:  SCOPE
1830  012104  012737  012236  002376          MOV     #MRET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
1831  012112  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1832  012120  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
1833  012124  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
1834  012130  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
1835  012134  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
1836  012140  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
1837  012144  004737  023342                  JSR     PC,RANDU2       ;GET RANDOM INPUT DATA
1838  012150  002426  002436                  .WORD   LONUM,HINUM     ;
1839  012154  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
1840  012160  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1841  012164  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1842  012170  025244                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
1843  012172  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1844
1845  012176  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1846  012202  170127  040000                  LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
1847  012206  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1848  012212  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1849  012216  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1850  012222  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
1851  012226  012737  012234  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1852
1853                                          ;*************************************************************
1854
1855  012234  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1856  012236  171201                  MRET3:  MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
1857  012240  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
```



```
1858  012244  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
1859  012252  001403                          BEQ     MTST3           ;BRANCH IF OK
1860  012254  170237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1861  012260  104003                          ERROR   3               ;FPS ERROR
1862
1863  012262  173702                  MTST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1864  012264  170003                          CFCC                    ;COPY FLOATING CONDITION CODES
1865  012266  001416                          BEQ     MEND3           ;ANSWERS CHECK
1866                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
1867  012270  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1868  012274  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1869  012302  005637  002406                  SBC     ANS1
1870  012306  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
1871  012312  170003                          CFCC                    ;COPY FLOATING CONDITION CODES
1872  012314  001403                          BEQ     MEND3           ;BRANCH IF OK
1873  012316  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
1874  012322  104011                          ERROR   11              ;FPU AND FORTRAN DISAGREE
1875
1876  012324  005037  002362          MEND3:  CLR     FPS             ;CLEAR FPP FPS BUFFER
1877
1878
1879
1880
1881                                          ;;*************************************************************
1882                                  ;*TEST 24        EXERCISE MULD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
1883                                          ;;*************************************************************
1884  012330  000004                  TST24:  SCOPE
1885  012332  012737  012464  002376          MOV     #MRET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
1886  012340  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1887  012346  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
1888  012352  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
1889  012356  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
1890  012362  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
1891  012366  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
1892  012372  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1893  012376  002426  002436                  .WORD   LONUM,HINUM     ;
1894  012402  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
1895  012406  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1896  012412  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1897  012416  025244                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
1898  012420  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1899
1900  012424  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1901  012430  170127  040200                  LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
1902  012434  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1903  012440  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1904  012444  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1905  012450  170127  004600                  LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
1906  012454  012737  012462  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1907
1908                                          ;*************************************************************
1909
1910  012462  172680                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1911  012464  171201                  MRET4:  MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
1912  012466  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1913  012472  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
```

```
 1914  012500  001403                          BEQ     MTST4           ;BRANCH IF OK
 1915  012502  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 1916  012506  104021                          ERROR   21              ;FPS ERROR
 1917
 1918  012510  173702                  MTST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
 1919  012512  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1920  012514  001422                          BEQ     MEND4           ;ANSWERS CHECK
 1921                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
 1922  012516  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 1923  012522  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
 1924  012530  005637  002412                  SBC     ANS1+4
 1925  012534  005637  002410                  SBC     ANS1+2
 1926  012540  005637  002406                  SBC     ANS1
 1927  012544  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1928  012550  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1929  012552  001403                          BEQ     MEND4           ;BRANCH IF OK
 1930  012554  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 1931  012560  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
 1932
 1933  012562  005037  002362          MEND4:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 1934
 1935
 1936
 1937                                          ;;*****************************************************************
 1938                                          ;*TEST 25        EXERCISE MULF, ALL INTERRUPTS ON, TRUNCATE MODE
 1939                                          ;;*****************************************************************
 1940  012566  000004                  TST25:  SCOPE
 1941  012570  012737  012722  002376          MOV     $MRET5,EXPFPA   ;ADDR OF INSTR BEING TESTED
 1942  012576  012737  007440  002400          MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
 1943  012604  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
 1944  012610  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
 1945  012614  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
 1946  012620  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
 1947  012624  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
 1948  012630  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
 1949  012634  002426  002436                  .WORD   LONUM,HINUM     ;
 1950  012640  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
 1951  012644  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
 1952  012650  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
 1953  012654  025244                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
 1954  012656  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
 1955
 1956  012662  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
 1957  012666  170127  040000                  LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
 1958  012672  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
 1959  012676  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
 1960  012702  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
 1961  012706  170127  007440                  LDFPS   #007440         ;TURN INTERRUPTS ON
 1962  012712  012737  012720  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
 1963
 1964                                          ;*************************************************************
 1965
 1966  012720  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
 1967  012722  171201                  MRET5:  MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
 1968  012724  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
 1969  012730  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
```



```
 1970  012736  001403                          BEQ     MERR5           ;BRANCH IF OK
 1971  012740  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 1972  012744  104003                          ERROR   3               ;FPS ERROR
 1973
 1974  012746  005737  002400          MERR5:  TST     $FPS            ;ERROR BIT SET ?
 1975  012752  100014                          BPL     MTST5           ;NO, DONT GET FEC/FEA
 1976  012754  170337  002364                  STST    FEC             ;YES, CHECK STATUS
 1977
 1978  012760  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
 1979  012766  001401                          BEQ     1$              ;BRANCH IF OK
 1980  012770  104025                          ERROR   25              ;FEC IS WRONG
 1981
 1982  012772  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
 1983  013000  001401                          BEQ     MTST5           ;BRANCH IF OK
 1984  013002  104025                          ERROR   25              ;WRONG ADDRESS IN FEA
 1985
 1986  013004  173702                  MTST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
 1987  013006  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1988  013010  001427                          BEQ     MEND5           ;ANSWERS CHECK
 1989                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
 1990  013012  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 1991  013016  062737  000001  002410          ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
 1992  013024  005537  002406                  ADC     ANS1
 1993  013030  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1994  013034  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1995  013036  001414                          BEQ     MEND5           ;BRANCH IF OK
 1996  013040  162737  000002  002410          SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 1997  013046  005637  002406                  SBC     ANS1
 1998  013052  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1999  013056  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2000  013060  001403                          BEQ     MEND5           ;BRANCH IF OK
 2001  013062  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 2002  013066  104011                          ERROR   11              ;FPU AND FORTRAN DISAGREE
 2003
 2004  013070  005037  002362          MEND5:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2005
 2006
 2007
 2008                                          ;;*****************************************************************
 2009                                          ;*TEST 26        EXERCISE MULD, ALL INTERRUPTS ON, TRUNCATE MODE
 2010                                          ;;*****************************************************************
 2011                                  TST26:  SCOPE
 2012  013074  000004                          MOV     #MRET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
 2013  013076  012737  013230  002376          MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
 2014  013104  012737  007640  002400          CLR     $FEC            ;CLR FORTRAN FEC
 2015  013112  005037  002402                  CLR     $FEA            ;CLR FORTRAN FEA
 2016  013116  005037  002404                  CLR     FPS             ;CLR FPU FPS BUFFER
 2017  013122  005037  002362                  CLR     FEC             ;CLR FPU FEC BUFFER
 2018  013126  005037  002364                  CLR     FEA             ;CLR FPU FEA BUFFER
 2019  013132  005037  002366                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
 2020  013136  004737  023332                  .WORD   LONUM,HINUM     ;
 2021  013142  002426  002436                  JSR     R4,$POLSH       ;ENTER POLISH MODE
 2022  013146  004437  023506                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
 2023  013152  023510  002426                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
 2024  013156  023510  002436                  $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
 2025  013162  025244
```

```
2026  013164  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2027
2028  013170  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2029  013174  170127  040200                  LDFPS   #40200          ;SET FD OF FPS ONLY, INTERRUPT DISABLE
2030  013200  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2031  013204  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2032  013210  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2033  013214  170127  007640                  LDFPS   #7640           ;TURN INTERRUPTS ON
2034  013220  012737  013226  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2035
2036                                          ;*********************************************************.
2037
2038  013226  172600                  MRET6:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2039  013230  171201                          MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
2040  013232  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2041  013236  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2042  013244  001403                          BEQ     MERR6           ;BRANCH IF OK
2043  013246  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2044  013252  104021                          ERROR   21              ;FPS ERROR
2045
2046  013254  005737  002400          MERR6:  TST     $FPS            ;ERROR BIT SET ?
2047  013260  100014                          BPL     MTST6           ;NO, DONT GET FEC/FEA
2048  013262  170337  002364                  STST    FEC             ;YES, CHECK STATUS
2049
2050  013266  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2051  013274  001401                          BEQ     1$              ;BRANCH IF OK
2052  013276  104031                          ERROR   31              ;FEC IS WRONG
2053
2054  013300  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2055  013306  001401                          BEQ     MTST6           ;BRANCH IF OK
2056  013310  104031                          ERROR   31              ;WRONG ADDRESS IN FEA
2057
2058  013312  173702                  MTST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2059  013314  170000                          CFCC                    ;COPY FLOATING CONDITION CODES '
2060  013316  001437                          BEQ     MEND6           ;ANSWERS CHECK
2061                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
2062  013320  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2063  013324  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
2064  013332  005537  002412                  ADC     ANS1+4
2065  013336  005537  002410                  ADC     ANS1+2
2066  013342  005537  002406                  ADC     ANS1
2067  013346  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2068  013352  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2069  013354  001420                          BEQ     MEND6           ;BRANCH IF OK
2070  013356  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
2071  013364  005637  002412                  SBC     ANS1+4
2072  013370  005637  002410                  SBC     ANS1+2
2073  013374  005637  002406                  SBC     ANS1
2074  013400  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2075  013404  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2076  013406  001401                          BEQ     MEND6           ;BRANCH IF OK
2077  013410  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2078  013414  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
2079
2080  013416  005037  002362          MEND6:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2081
```



```
2082
2083
2084
2085                                          ;;*****************************************************************
2086                                          ;*TEST 27        EXERCISE MULF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
2087                                          ;;*****************************************************************
2088  013422  000004                  TST27:  SCOPE
2089  013424  012737  013556  002376          MOV     #MRET7,FXPFEA   ;ADDR OF INSTR BEING TESTED
2090  013432  012737  004440  002400          MOV     #^O4440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2091  013440  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2092  013444  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2093  013450  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2094  013454  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2095  013460  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2096  013464  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2097  013470  002426  002436                  .WORD   LONUM,HINUM     ;
2098  013474  004437  023506                  JSH     R4,$POLSH       ;ENTER POLISH MODE
2099  013500  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2100  013504  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2101  013510  025244                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
2102  013512  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2103
2104  013516  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2105  013522  170127  040000                  LDFPS   #40000          ;CLEAR THE FPS, INTERRUPT DISABLE
2106  013526  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2107  013532  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2108  013536  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2109  013542  170127  004440                  LDFPS   #4440           ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
2110  013546  012737  013554  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2111
2112                                          ;*********************************************************
2113
2114  013554  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2115  013556  171201                  MRET7:  MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
2116  013560  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2117  013564  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2118  013572  001403                          BEQ     MTST7           ;BRANCH IF OK
2119  013574  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2120  013600  104003                          ERROR   3               ;FPS ERROR
2121
2122  013602  173702                  MTST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2123  013604  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2124  013606  001427                          BEQ     MEND7           ;ANSWERS CHECK
2125                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
2126  013610  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2127  013614  062737  000001  002410          ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
2128  013622  005537  002406                  ADC     ANS1
2129  013626  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2130  013632  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2131  013634  001414                          BEQ     MEND7           ;BRANCH IF OK
2132  013636  162737  000002  002410          SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
2133  013644  005637  002406                  SBC     ANS1
2134  013650  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2135  013654  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2136  013656  001403                          BEQ     MEND7           ;BRANCH IF OK
2137  013660  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
```

```
2138  013664  104011                        ERROR   11              ;FPU AND FORTRAN DISAGREE
2139
2140  013666  005037  002362          MEND7:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2141
2142
2143
2144
2145                                  ;;*************************************************************
2146                                  ;*TEST 30        EXERCISE MULD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
2147                                  ;;*************************************************************
2148  013672  000004                  TST30:  SCOPE
2149  013674  012737  014026  002376          MOV     #MRET10,EXPEA   ;ADDR OF INSTR BEING TESTED
2150  013702  012737  004640  002400          MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2151  013710  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2152  013714  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2153  013720  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2154  013724  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2155  013730  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2156  013734  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2157  013740  002426  002436                  .WORD   LONUM,HINUM     ;
2158  013744  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2159  013750  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2160  013754  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2161  013760  025244                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
2162  013762  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2163
2164  013766  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2165  013772  170127  040200                  LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
2166  013776  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2167  014002  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2168  014006  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2169  014012  170127  004640                  LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
2170  014016  012737  014024  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2171
2172                                  ;*************************************************************
2173
2174  014024  172680                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2175  014026  171201          MRET10: MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
2176  014030  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2177  014034  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2178  014042  001403                          BEQ     MTST10          ;BRANCH IF OK
2179  014044  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2180  014050  104021                          ERROR   21              ;FPS ERROR
2181
2182  014052  173702          MTST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2183  014054  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2184  014056  001437                          BEQ     MEND10          ;ANSWERS CHECK
2185                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
2186  014060  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2187  014064  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
2188  014072  005537  002412                  ADC     ANS1+4
2189  014076  005537  002410                  ADC     ANS1+2
2190  014102  005537  002406                  ADC     ANS1
2191  014106  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2192  014112  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2193  014114  001420                          BEQ     MEND10          ;BRANCH IF OK
```



```
2194  014116  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
2195  014124  005637  002412                  SBC     ANS1+4
2196  014130  005637  002410                  SBC     ANS1+2
2197  014134  005637  002406                  SBC     ANS1
2198  014140  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2199  014144  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2200  014146  001403                          BEQ     MEND10          ;BRANCH IF OK
2201  014150  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2202  014154  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
2203
2204  014156  005037  002362          MEND10: CLR     FPS             ;CLEAR FPP FPS BUFFER
2205
```

```
                                ;;*****************************************************************
                                ;*TEST 31        EXERCISE DIVF, ALL INTERRUPTS ON, ROUNDING MODE
                                ;;*****************************************************************
014162 000004                   TST31:  SCOPE
014164 012737 014316 002376             MOV     #DRET1,@XPFEA   ;ADDR OF INSTR BEING TESTED
014172 012737 007400 002400             MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
014200 005037 002402                    CLR     $FEC            ;CLR FORTRAN FEC
014204 005037 002404                    CLR     $FEA            ;CLR FORTRAN FEA
014210 005037 002362                    CLR     FPS             ;CLR FPU FPS BUFFER
014214 005037 002364                    CLR     FEC             ;CLR FPU FEC BUFFER
014220 005037 002366                    CLR     FEA             ;CLR FPU FEA BUFFER
014224 004737 023342                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
014230 002426 002436                    ,WORD   LONUM,HINUM     ;
014234 004437 023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE
014240 023510 002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
014244 023510 002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
014250 026364                           $DIV                    ;ADDRESS OF FORTRAN DIVIDE
014252 023540 002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE

014256 013700 002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
014262 170127 040000                    LDFPS   #040000         ;SET INTERRUPT DISABLE
014266 172437 002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
014272 172537 002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
014276 172737 002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM '
014302 170127 007400                    LDFPS   #007400         ;TURN INTERRUPTS ON
014306 012737 014314 001110             MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

014314 172600                   DRET1:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
014316 174601                           DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
014320 170237 002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
014324 023737 002362 002400             CMP     FPS,$FPS        ;CHECK FPS
014332 001403                           BEQ     DERR1           ;BRANCH IF OK
014334 174237 002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
014340 104016                           ERROR   16              ;FPS ERROR

014342 005737 002400            DERR1:  TST     $FPS            ;ERROR BIT SET ?
014346 100014                           BPL     DTST1           ;NO, DONT GET FEC/FEA
014350 170337 002364                    STST    FEC             ;YES, CHECK STATUS

014354 023737 002364 002402             CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
014362 001401                           BEQ     1$              ;BRANCH IF OK
014364 104026                           ERROR   26              ;FEC IS WRONG

014366 023737 002366 002404     1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
014374 001401                           BEQ     DTST1           ;BRANCH IF OK
014376 104026                           ERROR   26              ;WRONG ADDRESS IN FEA

014400 173702                   DTST1:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
014402 170000                           CFCC                    ;COPY FLOATING CONDITION CODES
014404 001416                           BEQ     DEND1           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
014406 174237 002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
014412 162737 000001 002410             SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
014420 005637 002406                    SBC     ANS1
```



```
014424 173737 002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
014430 170000                           CFCC                    ;COPY FLOATING CONDITION CODES
014432 001403                           BEQ     DEND1           ;BRANCH IF OK
014434 174237 002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
014440 104013                           ERROR   13              ;FPU AND FORTRAN DISAGREE

014442 005037 002362            DEND1:  CLR     FPS             ;CLEAR FPP FPS BUFFER


                                ;;*****************************************************************
                                ;*TEST 32        EXERCISE DIVD, ALL INTERRUPTS ON, ROUNDING MODE
                                ;;*****************************************************************
014446 000004                   TST32:  SCOPE
014450 012737 014602 002376             MOV     #DRET2,@XPFEA   ;ADDR OF INSTR BEING TESTED
014456 012737 007600 002400             MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
014464 005037 002402                    CLR     $FEC            ;CLR FORTRAN FEC
014470 005037 002404                    CLR     $FEA            ;CLR FORTRAN FEA
014474 005037 002362                    CLR     FPS             ;CLR FPU FPS BUFFER
014500 005037 002364                    CLR     FEC             ;CLR FPU FEC BUFFER
014504 005037 002366                    CLR     FEA             ;CLR FPU FEA BUFFER
014510 004737 023332                    JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
014514 002426 002436                    ,WORD   LONUM,HINUM     ;
014520 004437 023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE
014524 023510 002426                    $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
014530 023510 002436                    $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
014534 026364                           $DIV                    ;ADDRESS OF FORTRAN DIVIDE
014536 023540 002416                    $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

014542 013700 002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
014546 170127 040200                    LDFPS   #040200         ;SET FID AND FD
014552 172437 002426                    LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
014556 172537 002436                    LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
014562 172737 002416                    LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
014566 170127 007600                    LDFPS   #007600         ;TURN INTERRUPTS ON
014572 012737 014600 001110             MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

014600 172600                   DRET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
014602 174601                           DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
014604 170237 002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
014610 023737 002362 002400             CMP     FPS,$FPS        ;CHECK FPS
014616 001403                           BEQ     DERR2           ;BRANCH IF OK
014620 174237 002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
014624 104022                           ERROR   22              ;FPS ERROR

014626 005737 002400            DERR2:  TST     $FPS            ;ERROR BIT SET ?
014632 100014                           BPL     DTST2           ;NO, DONT GET FEC/FEA
014634 170337 002364                    STST    FEC             ;YES, CHECK STATUS

014640 023737 002364 002402             CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
014646 001401                           BEQ     1$              ;BRANCH IF OK
014650 104032                           ERROR   32              ;FEC IS WRONG
```

```
2318  014652  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2319  014660  001401                          BEQ     DTST2           ;BRANCH IF OK
2320  014662  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
2321
2322  014664  173702                  DTST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2323  014666  170003                          CFCC                    ;COPY FLOATING CONDITION CODES
2324  014670  001422                          BEQ     DEND2           ;ANSWERS CHECK
2325                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
2326  014672  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2327  014676  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
2328  014704  005637  002412                  SBC     ANS1+4
2329  014710  005637  002410                  SBC     ANS1+2
2330  014714  005637  002406                  SBC     ANS1
2331  014720  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2332  014724  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2333  014726  001403                          BEQ     DEND2           ;BRANCH IF OK
2334  014730  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2335  014734  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
2336
2337  014736  005037  002362          DEND2:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2338
2339
2340
2341
2342                                          ;;*****************************************************************
2343                                          ;*TEST 33        EXERCISE DIVF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
2344                                          ;;*****************************************************************
2345  014742  000004          TST33:  SCOPE
2346  014744  012737  015076  002376          MOV     #DRET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
2347  014752  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2348  014760  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2349  014764  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2350  014770  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2351  014774  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2352  015000  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2353  015004  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2354  015010  002426  002436                  .WORD   LONUM,HINUM     ;
2355  015014  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2356  015020  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2357  015024  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2358  015030  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2359  015032  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2360
2361  015036  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2362  015042  170127  040000                  LDFPS   #040000         ;SET FID
2363  015046  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2364  015052  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2365  015056  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2366  015062  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
2367  015066  012737  015074  001110          MOV     8.,+6,$LPADR    ;RESET LOOP ADDRESS
2368
2369                                          ;*************************************************************
2370
2371  015074  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2372  015076  174601                  DRET3:  DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
2373  015100  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
```



```
2374  015104  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2375  015112  001403                          BEQ     DERR3           ;BRANCH IF OK
2376  015114  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2377  015120  104016                          ERROR   16              ;FPS ERROR
2378
2379  015122  005737  002400          DERR3:  TST     $FPS            ;ERROR BIT SET ?
2380  015126  100014                          BPL     DTST3           ;NO, DONT GET FEC/FEA
2381  015130  170337  002364                  STST    FEC             ;YES, CHECK STATUS
2382
2383  015134  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2384  015142  001401                          BEQ     1$              ;BRANCH IF OK
2385  015144  104026                          ERROR   26              ;FEC IS WRONG
2386
2387  015146  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2388  015154  001401                          BEQ     DTST3           ;BRANCH IF OK
2389  015156  104026                          ERROR   26              ;WRONG ADDRESS IN FEA
2390
2391  015160  173702                  DTST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2392  015162  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2393  015164  001416                          BEQ     DEND3           ;ANSWERS CHECK
2394                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
2395  015166  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2396  015172  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
2397  015200  005637  002406                  SBC     ANS1
2398  015204  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2399  015210  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2400  015212  001403                          BEQ     DEND3           ;BRANCH IF OK
2401  015214  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2402  015220  104013                          ERROR   13              ;FPU AND FORTRAN DISAGREE
2403
2404  015222  005037  002362          DEND3:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2405
2406
2407
2408
2409                                          ;;*****************************************************************
2410                                          ;*TEST 34        EXERCISE DIVD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
2411                                          ;;*****************************************************************
2412  015226  000004          TST34:  SCOPE
2413  015230  012737  015362  002376          MOV     #DRET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
2414  015236  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2415  015244  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2416  015250  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2417  015254  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2418  015260  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2419  015264  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2420  015270  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2421  015274  002426  002436                  .WORD   LONUM,HINUM     ;
2422  015300  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2423  015304  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2424  015310  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2425  015314  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2426  015316  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2427
2428  015322  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2429  015326  170127  040200                  LDFPS   #040200         ;SET FID AND FD
```

```
2430  015332  172437  002126          LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2431  015336  172537  002436          LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2432  015342  172737  002416          LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2433  015346  170127  004600          LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
2434  015352  012737  015360  001110  MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2435
2436                                  ;*************************************************************
2437
2438  015360  172600                  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2439  015362  174601          DRET4:  DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
2440  015364  170237  002362          STFPS   FPS             ;STORE FLOATING POINT STATUS
2441  015370  023737  002362  002400  CMP     FPS,$FPS        ;CHECK FPS
2442  015376  001403                  BEQ     DERR4           ;BRANCH IF OK
2443  015400  174237  002406          STD     AC2,ANS1        ;SAVE FPU ANSWER
2444  015404  104022                  ERROR   22              ;FPS ERROR
2445
2446  015406  005737  002400  DERR4:  TST     $FPS            ;ERROR BIT SET ?
2447  015412  100014                  BPL     DTST4           ;NO, DONT GET FEC/FEA
2448  015414  170337  002364          STST    FEC             ;YES, CHECK STATUS
2449
2450  015420  023737  002364  002402  CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2451  015426  001401                  BEQ     1$              ;BRANCH IF OK
2452  015430  104032                  ERROR   32              ;FEC IS WRONG
2453
2454  015432  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2455  015440  001401                  BEQ     DTST4           ;BRANCH IF OK
2456  015442  104032                  ERROR   32              ;WRONG ADDRESS IN FEA
2457
2458  015444  173702          DTST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2459  015446  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2460  015450  001422                  BEQ     DEND4           ;ANSWERS CHECK
                                      ;COMPENSATE FOR FORTRAN INACCURACIES.
2462  015452  174237  002406          STD     AC2,ANS1        ;SAVE FPU ANSWER
2463  015456  162737  000001  002414  SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
2464  015464  005637  002412          SBC     ANS1+4
2465  015470  005637  002410          SBC     ANS1+2
2466  015474  005637  002406          SBC     ANS1
2467  015500  173737  002406          CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2468  015504  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2469  015506  001403                  BEQ     DEND4           ;BRANCH IF OK
2470  015510  174237  002406          STD     AC2,ANS1        ;SAVE FPU ANSWER
2471  015514  104014                  ERROR   14              ;FPU AND FORTRAN DISAGREE
2472
2473  015516  005037  002362  DEND4:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2474
2475
2476
2477
2478                                  ;;*************************************************************
2479                                  ;*TEST 35        EXERCISE DIVF, ALL INTERRUPTS ON, TRUNCATE MODE
2480                                  ;*************************************************************
2481  015522  000004          TST35:  SCOPE
2482  015524  012737  015656  002376  MOV     #DRET5,EXPFEA   ;ADDR OF INSTR BEING TESTED
2483  015532  012737  007440  002400  MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2484  015540  005037  002402          CLR     $FEC            ;CLR FORTRAN FEC
2485  015544  005037  002404          CLR     $FEA            ;CLR FORTRAN FEA
```



```
2486  015550  005037  002362          CLR     FPS             ;CLR FPU FPS BUFFER
2487  015554  005037  002364          CLR     FEC             ;CLR FPU FEC BUFFER
2488  015560  005037  002366          CLR     FEA             ;CLR FPU FEA BUFFER
2489  015564  004737  023342          JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2490  015570  002426  002436          .WORD   LONUM,HINUM     ;
2491  015574  004437  023506          JSR     R4,$POLSH       ;ENTER POLISH MODE
2492  015600  023516  002426          $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2493  015604  023516  002436          $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2494  015610  026364                  $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2495  015612  023540  002416          $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2496
2497  015616  013700  002400          MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2498  015622  170127  040000          LDFPS   #040000         ;SET FID
2499  015626  172437  002426          LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2500  015632  172537  002436          LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2501  015636  172737  002416          LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2502  015642  170127  007440          LDFPS   #007440         ;TURN INTERRUPTS ON
2503  015646  012737  015654  001110  MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2504
2505                                  ;*************************************************************
2506
2507  015654  172600                  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2508  015656  174601          DRET5:  DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
2509  015660  170237  002362          STFPS   FPS             ;STORE FLOATING POINT STATUS
2510  015664  023737  002362  002400  CMP     FPS,$FPS        ;CHECK FPS
2511  015672  001403                  BEQ     DERR5           ;BRANCH IF OK
2512  015674  174237  002406          STF     AC2,ANS1        ;SAVE FPU ANSWER
2513  015700  104016                  ERROR   16              ;FPS ERROR
2514
2515  015702  005737  002400  DERR51: TST     $FPS            ;ERROR BIT SET ?
2516  015706  100014                  BPL     DTST5           ;NO, DONT GET FEC/FEA
2517  015710  170337  002364          STST    FEC             ;YES, CHECK STATUS
2518
2519  015714  023737  002364  002402  CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2520  015722  001401                  BEQ     1$              ;BRANCH IF OK
2521  015724  104026                  ERROR   26              ;FEC IS WRONG
2522
2523  015726  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2524  015734  001401                  BEQ     DTST5           ;BRANCH IF OK
2525  015736  104026                  ERROR   26              ;WRONG ADDRESS IN FEA
2526
2527  015740  173702          DTST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2528  015742  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2529  015744  001427                  BEQ     DEND5           ;ANSWERS CHECK
                                      ;COMPENSATE FOR FORTRAN INACCURACIES.
2531  015746  174237  002406          STF     AC2,ANS1        ;SAVE FPU ANSWER
2532  015752  062737  000001  002410  ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
2533  015760  005537  002406          ADC     ANS1
2534  015764  173737  002406          CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2535  015770  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2536  015772  001414                  BEQ     DEND5           ;BRANCH IF OK
2537  015774  162737  000002  002410  SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
2538  016002  005637  002406          SBC     ANS1
2539  016006  173737  002406          CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2540  016012  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2541  016014  001403                  BEQ     DEND5           ;BRANCH IF OK
```

```
2542  016016  174237  002406          STF     AC2,ANS1        ;SAVE FPU ANSWER
2543  016022  104013                  ERROR   13              ;FPU AND FORTRAN DISAGREE
2544
2545  016024  005037  002362  DEND5:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2546
2547
2548
2549
2550                                  ;;*****************************************************************
2551                                  ;*TEST 36        EXERCISE DIVD, ALL INTERRUPTS ON, TRUNCATE MODE
2552                                  ;;*****************************************************************
2553  016030  000004          TST36:  SCOPE
2554  016032  012737  016164  002376          MOV     $DRET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
2555  016040  012737  007640  002400          MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2556  016046  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2557  016052  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2558  016056  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2559  016062  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2560  016066  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2561  016072  004737  023332                  JSP     PC,RANDL4       ;GET RANDOM INPUT DATA
2562  016076  002426  002436                  .WORD   LONUM,HINUM     ;
2563  016102  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2564  016106  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2565  016112  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2566  016116  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2567  016120  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2568
2569  016124  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2570  016130  170127  040200                  LDFPS   #04#200         ;SET FID AND FD
2571  016134  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2572  016140  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2573  016144  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2574  016150  170127  007640                  LDFPS   #007640         ;TURN INTERRUPTS ON
2575  016154  012737  016162  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2576
2577                                          ;*************************************************************
2578
2579  016162  172600                  DRET6:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2580  016164  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
2581  016166  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2582  016172  027737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2583  016200  001403                          BEQ     DERR6           ;BRANCH IF OK
2584  016202  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2585  016206  104022                          ERROR   22              ;FPS ERROR
2586
2587  016210  005737  002400          DERR6:  TST     $FPS            ;ERROR BIT SET ?
2588  016214  100014                          BPL     DTST6           ;NO, DONT GET FEC/FEA
2589  016216  170337  002364                  STST    FEC             ;YES, CHECK STATUS
2590
2591  016222  027737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2592  016230  001401                          BEQ     1$              ;BRANCH IF OK
2593  016232  104032                          ERROR   32              ;FEC IS WRONG
2594
2595  016234  027737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2596  016242  001401                          BEQ     DTST6           ;BRANCH IF OK
2597  016244  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
```



```
2598
2599  016246  173702          DTST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2600  016250  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2601  016252  001437                  BEQ     DEND6           ;ANSWERS CHECK
                                      ;COMPENSATE FOR FORTRAN INACCURACIES,
2602
2603  016254  174237  002406          STD     AC2,ANS1        ;SAVE FPU ANSWER
2604  016260  062737  000001  002414  ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
2605  016266  005537  002412          ADC     ANS1+4
2606  016272  005537  002410          ADC     ANS1+2
2607  016276  005537  002406          ADC     ANS1
2608  016302  173737  002406          CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2609  016306  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2610  016310  001420                  BEQ     DEND6           ;BRANCH IF OK
2611  016312  162737  000002  002414  SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
2612  016320  005637  002412          SBC     ANS1+4
2613  016324  005637  002410          SBC     ANS1+2
2614  016330  005637  002406          SBC     ANS1
2615  016334  173737  002406          CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2616  016340  170000                  CFCC                    ;COPY FLOATING CONDITION CODES
2617  016342  001403                  BEQ     DEND6           ;BRANCH IF OK
2618  016344  174237  002406          STD     AC2,ANS1        ;SAVE FPU ANSWER
2619  016350  104014                  ERROR   14              ;FPU AND FORTRAN DISAGREE
2620
2621  016352  005037  002362  DEND6:  CLR     FPS             ;CLEAR FPP FPS BUFFER
2622
2623
2624
2625
2626                                  ;;*****************************************************************
2627                                  ;*TEST 37        EXERCISE DIVF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
2628                                  ;;*****************************************************************
2629  016356  000004          TST37:  SCOPE
2630  016360  012737  016512  002376          MOV     #DRET7,EXPFEA   ;ADDR OF INSTR BEING TESTED
2631  016366  012737  004440  002400          MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2632  016374  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2633  016400  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2634  016404  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2635  016410  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2636  016414  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2637  016420  004737  023342                  JSP     PC,RANDL2       ;GET RANDOM INPUT DATA
2638  016424  002426  002436                  .WORD   LONUM,HINUM     ;
2639  016430  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2640  016434  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2641  016440  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2642  016444  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2643  016446  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2644
2645  016452  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2646  016456  170127  040000                  LDFPS   #040000         ;SET FID
2647  016462  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2648  016466  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2649  016472  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2650  016476  170127  004440                  LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
2651  016502  012737  016510  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2652
2653                                          ;*************************************************************
```

```
                                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
016512  174001                 DRE77:  DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
016514  170237  002362                 STFPS   FPS             ;STORE FLOATING POINT STATUS
016520  023737  023737  002400         CMP     FPS,$FPS        ;CHECK FPS
016526  001403                         BEQ     DERR7           ;BRANCH IF OK
016530  174237  002406                 STF     AC2,ANS1        ;SAVE FPU ANSWER
016534  104016                         ERROR   16              ;FPS ERROR
016536  005737  002400         DERR7:  TST     $FPS            ;ERROR BIT SET 7
016542  100014                         BPL     DTST7           ;NO, DONT GET FEC/FEA
016544  170337  002364                 STST    FEC             ;YES, CHECK STATUS
016550  073737  002364  002402         CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
016556  001401                         BEQ     1$              ;BRANCH IF OK
016560  104026                         ERROR   26              ;FEC 1S WRONG
016562  023737  002366  002404  1$:    CMP     FEA,$FEA        ;CHECK FLOATING PC
016570  001401                         BEQ     DTST7           ;BRANCH IF OK
016572  104026                         ERROR   26              ;WRONG ADDRESS IN FEA
016574  173702                 DTST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
016576  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
016600  001427                         BEQ     DEND7           ;ANSWERS CHECK
                                       ;COMPENSATE FOR FORTRAN INACCURACIES.
016602  174237  002406                 STF     AC2,ANS1        ;SAVE FPU ANSWER
016606  062737  000001  002410         ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
016614  005537  002406                 ADC     ANS1
016620  173737  002406                 CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
016624  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
016626  001414                         BEQ     DEND7           ;BRANCH IF OK
016630  162737  000002  002410         SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
016636  005637  002406                 SBC     ANS1
016642  173737  002406                 CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
016646  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
016650  001403                         BEQ     DEND7           ;BRANCH IF OK
016652  174237  002406                 STF     AC2,ANS1        ;SAVE FPU ANSWER
016656  104013                         ERROR   13              ;FPU AND FORTRAN DISAGREE
016660  005037  002362         DEND7:  CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                       ;;*************************************************************
                                       ;*TEST 40        EXERCISE DIVD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                       ;;*************************************************************
016664  000004                 TST40:  SCOPE
016666  012737  017020  002376         MOV     #DRET10,EXPFEA  ;ADDR OF INSTR BEING TESTED
016674  012737  004640  002400         MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
016702  005037  002402                 CLR     $FEC            ;CLR FORTRAN FEC
016706  005037  002404                 CLR     $FEA            ;CLR FORTRAN FEA
016712  005037  002362                 CLR     FPS             ;CLR FPU FPS BUFFER
016716  005037  002364                 CLR     FEC             ;CLR FPU FEC BUFFER
016722  005037  002366                 CLR     FEA             ;CLR FPU FEA BUFFER
016726  004737  023332                 JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
```



```
016732  002426  002436                 .WORD   LONUM,HINUM     ;
016736  004437  023506                 JSR     R4,$POLSH       ;ENTER POLISH MODE
016742  023510  002426                 $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
016746  023510  002436                 $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
016752  026364                         $DIV                    ;ADDRESS OF FORTRAN DIVIDE
016754  023540  002416                 $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
016760  013700  002400                 MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
016764  170127  040200                 LDFPS   #040200         ;SET FID AND FD
016770  172437  002426                 LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
016774  172537  002436                 LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
017000  172737  002416                 LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
017004  170127  004640                 LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
017010  012737  017016  001110         MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
```

```
017016  172600                 DRET10: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
017020  174601                         DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
017022  170237  002362                 STFPS   FPS             ;STORE FLOATING POINT STATUS
017026  023737  002362  002400         CMP     FPS,$FPS        ;CHECK FPS
017034  001403                         BEQ     DERR10          ;BRANCH IF OK
017036  174237  002406                 STD     AC2,ANS1        ;SAVE FPU ANSWER
017042  104022                         ERROR   22              ;FPS ERROR
017044  005737  002400         DERR10: TST     $FPS            ;ERROR BIT SET 7
017050  100014                         BPL     DTST10          ;NO, DONT GET FEC/FEA
017052  170337  002364                 STST    FEC             ;YES, CHECK STATUS
017056  023737  002364  002402         CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
017064  001401                         BEQ     1$              ;BRANCH IF OK
017066  104032                         ERROR   32              ;FEC IS WRONG
017070  023737  002366  002404  1$:    CMP     FEA,$FEA        ;CHECK FLOATING PC
017076  001443                         BEQ     DEND10          ;BRANCH IF OK
017100  104032                         ERROR   32              ;WRONG ADDRESS IN FEA
017102  173702                 DTST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
017104  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
017106  001437                         BEQ     DEND10          ;ANSWERS CHECK
                                       ;COMPENSATE FOR FORTRAN INACCURACIES.
017110  174237  002406                 STD     AC2,ANS1        ;SAVE FPU ANSWER
017114  062737  000001  002414         ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
017122  005537  002412                 ADC     ANS1+4
017126  005537  002410                 ADC     ANS1+2
017132  005537  002406                 ADC     ANS1
017136  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
017142  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
017144  001428                         BEQ     DEND10          ;BRANCH IF OK
017146  162737  000002  002414         SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
017154  005637  002412                 SBC     ANS1+4
017160  005637  002410                 SBC     ANS1+2
017164  005637  002406                 SBC     ANS1
017170  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
017174  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
017176  001403                         BEQ     DEND10          ;BRANCH IF OK
```

```
2766  017200  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2767  017204  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
2768
2769  017206  005037  002362          DEND10: CLR     FPS             ;CLEAR FPP FPS BUFFER
2770
2771
2772
2773
2774                                  ;;*************************************************************
2775                                  ;*TEST 41        EXERCISE DIVF, INTERRUPT DISABLE SET, ROUNDING MODE
2776                                  ;;*************************************************************
2777  017212  000004          TST41:  SCOPE
2778  017214  012737  017346  002376          MOV     #DRET11,EXPFEA  ;ADDR OF INSTR BEING TESTED
2779  017222  012737  047400  002400          MOV     #047400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2780  017230  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2781  017234  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2782  017240  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2783  017244  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2784  017250  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2785  017254  004737  023342                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2786  017260  002426  002436                  .WORD   LONUM,HINUM     ;
2787  017264  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2788  017270  023510  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2789  017274  023510  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2790  017300  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2791  017302  023540  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2792
2793  017306  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2794  017312  170127  040000                  LDFPS   #040000         ;SET FID
2795  017316  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2796  017322  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2797  017326  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2798  017332  170127  047400                  LDFPS   #047400         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
2799  017336  012737  017344  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2800
2801                                  ;*************************************************************
2802
2803  017344  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2804  017346  174601          DRET11: DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
2805  017350  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2806  017354  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2807  017362  001403                          BEQ     DERR11          ;BRANCH IF OK
2808  017364  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2809  017370  104016                          ERROR   16              ;FPS ERROR
2810
2811  017372  005737  002400          DERR11: TST     $FPS            ;ERROR BIT SET ?
2812  017376  100014                          BPL     DTST11          ;NO, DONT GET FEC/FEA
2813  017400  170337  002364                  STST    FEC             ;YES, CHECK STATUS
2814
2815  017404  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2816  017412  001401                          BEQ     16              ;BRANCH IF OK
2817  017414  104026                          ERROR   26              ;FEC IS WRONG
2818
2819  017416  023737  002366  002404  16:      CMP     FEA,$FEA        ;CHECK FLOATING PC
2820  017424  001401                          BEQ     DTST11          ;BRANCH IF OK
2821  017426  104026                          ERROR   26              ;WRONG ADDRESS IN FEA
```



```
2822
2823  017430  173702                  DTST11: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2824  017432  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2825  017434  001416                          BEQ     DEND11          ;ANSWERS CHECK
2826                                          ;COMPENSATE FOR FORTRAN INACCURACIES.
2827  017436  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2828  017442  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
2829  017450  005637  002406                  SBC     ANS1
2830  017454  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2831  017460  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
2832  017462  001403                          BEQ     DEND11          ;BRANCH IF OK
2833  017464  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
2834  017470  104013                          ERROR   13              ;FPU AND FORTRAN DISAGREE
2835
2836  017472  005037  002362          DEND11: CLR     FPS             ;CLEAR FPP FPS BUFFER
2837
2838
2839
2840                                  ;;*************************************************************
2841                                  ;*TEST 42        EXERCISE DIVD, INTERRUPT DISABLE SET, ROUNDING MODE
2842                                  ;;*************************************************************
2843  017476  000004          TST42:  SCOPE
2844  017500  012737  017632  002376          MOV     #DRET12,EXPFEA  ;ADDR OF INSTR BEING TESTED
2845  017506  012737  047600  002400          MOV     #047600,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
2846  017514  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
2847  017520  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
2848  017524  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
2849  017530  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
2850  017534  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2851  017540  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2852  017544  002426  002436                  .WORD   LONUM,HINUM     ;
2853  017550  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2854  017554  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2855  017560  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2856  017564  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2857  017566  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2858
2859  017572  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2860  017576  170127  040200                  LDFPS   #040200         ;SET FID AND FD
2861  017602  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2862  017606  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2863  017612  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2864  017616  170127  047600                  LDFPS   #047600         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
2865  017622  012737  017630  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2866
2867                                  ;*************************************************************
2868
2869  017630  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2870  017632  174601          DRET12: DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
2871  017634  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2872  017640  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
2873  017646  001403                          BEQ     DERR12          ;BRANCH IF OK
2874  017650  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
2875  017654  104022                          ERROR   22              ;FPS ERROR
2876
2877  017656  005737  002400          DERR12: TST     $FPS            ;ERROR BIT SET ?
```

```
2878  017662  100014                        BPL     DTST12          ;NO, DONT GET FEC/FEA
2879  017664  170337  002364                STST    FEC             ;YES, CHECK STATUS
2880
2881  017670  023737  002364  002402        CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2882  017676  001401                        BEQ     1$              ;BRANCH IF OK
2883  017700  104032                        ERROR   32              ;FEC IS WRONG
2884
2885  017702  023737  002366  002404  1$:   CMP     FEA,$FEA        ;CHECK FLOATING PC
2886  017710  001401                        BEQ     DTST12          ;BRANCH IF OK
2887  017712  104032                        ERROR   32              ;WRONG ADDRESS IN FEA
2888
2889  017714  173702                DTST12: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2890  017716  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
2891  017720  001422                        BEQ     DEND12          ;ANSWERS CHECK
                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
2892
2893  017722  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
2894  017726  162737  000001  002414        SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
2895  017734  005637  002412                SBC     ANS1+4
2896  017740  005637  002410                SBC     ANS1+2
2897  017744  005637  002406                SBC     ANS1
2898  017750  173737  002406                CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2899  017754  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
2900  017756  001403                        BEQ     DEND12          ;BRANCH IF OK
2901  017760  174237  002406                STD     AC2,ANS1        ;SAVE FPU ANSWER
2902  017764  104014                        ERROR   14              ;FPU AND FORTRAN DISAGREE
2903
2904  017766  005037  002362          DEND12: CLR     FPS             ;CLEAR FPP FPS BUFFER
2905
2906
2907
2908
2909                                    ;;*****************************************************************
2910                                    ;*TEST 43        EXERCISE DIVF, INTERRUPT DISABLE SET, TRUNCATE MODE
2911                                    ;;*****************************************************************
2912  017772  000004                TST43:  SCOPE
2913  017774  012737  020126  002376        MOV     #DRET13,EXPFEA  ;ADDR OF INSTR BEING TESTED.
2914  020002  012737  047440  002400        MOV     #047440,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
2915  020010  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
2916  020014  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
2917  020020  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
2918  020024  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
2919  020030  005037  002366                CLR     FEA             ;CLR FPU FEA BUFFER
2920  020034  004737  023342                JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2921  020040  002426  002436                .WORD   LONUM,HINUM     ;
2922  020044  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE
2923  020050  023510  002426                $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2924  020054  023510  002436                $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2925  020060  026364                        $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2926  020062  023540  002416                $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2927
2928  020066  013700  002400                MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2929  020072  170127  040000                LDFPS   #040000         ;SET FID
2930  020076  172437  002426                LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2931  020102  172537  002436                LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2932  020106  172737  002416                LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2933  020112  170127  047440                LDFPS   #047440         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
```



```
2934  020116  012737  020124  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2935
2936                                    ;*************************************************************
2937
2938  020124  172600                        LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2939  020126  174601                DRET13: DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
2940  020130  170237  002362                STFPS   FPS             ;STORE FLOATING POINT STATUS
2941  020134  023737  002362  002400        CMP     FPS,$FPS        ;CHECK FPS
2942  020142  001403                        BEQ     DERR13          ;BRANCH IF OK
2943  020144  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
2944  020150  104016                        ERROR   16              ;FPS ERROR
2945
2946  020152  005737  002400          DERR13: TST     $FPS            ;ERROR BIT SET ?
2947  020156  100014                        BPL     DTST13          ;NO, DONT GET FEC/FEA
2948  020160  170337  002364                STST    FEC             ;YES, CHECK STATUS
2949
2950  020164  023737  002364  002402        CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2951  020172  001401                        BEQ     1$              ;BRANCH IF OK
2952  020174  104026                        ERROR   26              ;FEC IS WRONG
2953
2954  020176  023737  002366  002404  1$:   CMP     FEA,$FEA        ;CHECK FLOATING PC
2955  020204  001401                        BEQ     DTST13          ;BRANCH IF OK
2956  020206  104026                        ERROR   26              ;WRONG ADDRESS IN FEA
2957
2958  020210  173702                DTST13: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2959  020212  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
2960  020214  001427                        BEQ     DEND13          ;ANSWERS CHECK
                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
2961
2962  020216  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
2963  020222  162737  000001  002410        ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
2964  020230  005537  002406                ADC     ANS1
2965  020234  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2966  020240  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
2967  020242  001414                        BEQ     DEND13          ;BRANCH IF OK
2968  020244  162737  000002  002410        SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
2969  020252  005637  002406                SBC     ANS1
2970  020256  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2971  020262  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
2972  020264  001403                        BEQ     DEND13          ;BRANCH IF OK
2973  020266  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
2974  020272  104013                        ERROR   13              ;FPU AND FORTRAN DISAGREE
2975
2976  020274  005037  002362          DEND13: CLR     FPS             ;CLEAR FPP FPS BUFFER
2977
2978
2979
2980                                    ;;*****************************************************************
2981                                    ;*TEST 44        EXERCISE DIVD, INTERRUPT DISABLE SET, TRUNCATE MODE
2982                                    ;;*****************************************************************
2983  020300  000004                TST44:  SCOPE
2984  020302  012737  020434  002376        MOV     #DRET14,EXPFEA  ;ADDR OF INSTR BEING TESTED
2985  020310  012737  047640  002400        MOV     #047640,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
2986  020316  005037  002402                CLR     $FEC            ;CLR FORTRAN FEC
2987  020322  005037  002404                CLR     $FEA            ;CLR FORTRAN FEA
2988  020326  005037  002362                CLR     FPS             ;CLR FPU FPS BUFFER
2989  020332  005037  002364                CLR     FEC             ;CLR FPU FEC BUFFER
```

```
2990  020336  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
2991  020342  004737  023332                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2992  020346  002426  002436                  .WORD   LONUM,HINUM     ;
2993  020352  004437  023506                  JSR     R4,$POLSH       ;ENTER POLISH MODE
2994  020356  023510  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2995  020362  023510  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2996  020366  026364                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
2997  020370  023540  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2998
2999  020374  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
3000  020400  170127  040200                  LDFPS   #040200         ;SET FID AND FD
3001  020404  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
3002  020410  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
3003  020414  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
3004  020420  170127  047640                  LDFPS   #047640         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
3005  020424  012737  020432  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
3006
3007                                          ;*************************************************************
3008
3009  020432  172600                  DRET14: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
3010  020434  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
3011  020436  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
3012  020442  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
3013  020450  001403                          BEQ     DERR14          ;BRANCH IF OK
3014  020452  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
3015  020456  104022                          ERROR   22              ;FPS ERROR
3016
3017  020460  005737  002400          DERR14: TST     $FPS            ;ERROR BIT SET ?
3018  020464  100014                          BPL     DTST14          ;NO, DONT GET FEC/FEA
3019  020466  170337  002364                  STST    FEC             ;YES, CHECK STATUS
3020
3021  020472  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
3022  020500  001401                          BEQ     1$              ;BRANCH IF OK
3023  020502  104032                          ERROR   32              ;FEC IS WRONG
3024
3025  020504  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
3026  020512  001401                          BEQ     DTST14          ;BRANCH IF OK
3027  020514  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
3028
3029  020516  173702                  DTST14: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
3030  020520  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
3031  020522  001437                          BEQ     DEND14          ;ANSWERS CHECK
                                              ;COMPENSATE FOR FORTRAN INACCURACIES,
3032
3033  020524  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
3034  020530  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
3035  020536  005537  002412                  ADC     ANS1+4
3036  020542  005537  002410                  ADC     ANS1+2
3037  020546  005537  002406                  ADC     ANS1
3038  020552  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
3039  020556  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
3040  020560  001420                          BEQ     DEND14          ;BRANCH IF OK
3041  020562  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
3042  020570  005637  002412                  SBC     ANS1+4
3043  020574  005637  002410                  SBC     ANS1+2
3044  020600  005637  002406                  SBC     ANS1
3045  020604  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
```



```
3046  020610  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
3047  020612  001403                          BEQ     DEND14          ;BRANCH IF OK
3048  020614  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
3049  020620  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
3050
3051  020622  005037  002362          DEND14: CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
3052                                    ;;*************************************************************
3053                                    ;*TEST 45        ADDF, SUBF, MULF, DIVF EXERCISER
3054                                    ;;*************************************************************
3055  020626  000004                    TST45:  SCOPE
3056                                    ;*UNDERFLOW, OVERFLOW INTERRUPTS OFF; TRUNCATE MODE
3057
3058  020630  012737  000440  002400            MOV     #440,$FPS       ;SOFTWARE STATUS
3059  020636  005037  002402                    CLR     $FEC            ;CLEAR STATUS
3060  020642  005037  002404                    CLR     $FEA            ;
3061  020646  005037  002362                    CLR     FPS             ;
3062  020652  005037  002364                    CLR     FEC             ;
3063  020656  005037  002366                    CLR     FEA             ;
3064
3065  020662  004737  023342                    JSR     PC,RANDL2       ;GET 6 FLOAT RANDOM NUMBERS
3066  020666  002446  002476                    .WORD   OP1,OP4         ;
3067  020672  004737  023342                    JSR     PC,RANDL2       ;
3068  020676  002456  002506                    .WORD   OP2,OP5         ;
3069  020702  004737  023342            18:     JSR     PC,RANDL2       ;
3070  020706  002466  002516                    .WORD   OP3,OP6         ;
3071  020712  032737  077600  002516            BIT     #077600,OP6     ;LET'S NEVER DIVIDE BY ZERO
3072  020720  001770                    BEQ     18              ;EXPO OF OP6 IS ZERO, SO GET ANOTHER
3073
3074  020722  004437  023506                    JSR     R4,$POLSH       ;ENTER POLISH MODE TO CALCULATE:
3075  020726  023510  002446                    $PUSH   ,OP1            ;
3076  020732  023510  002456                    $PUSH   ,OP2            ;
3077  020736  023562                    $SUB                    ;  ANS2 = (OP1-OP2) * (OP3+OP4/OP6) * OP5
3078  020740  023510  002466                    $PUSH   ,OP3            ;
3079  020744  023510  002476                    $PUSH   ,OP4            ;
3080  020750  023510  002516                    $PUSH   ,OP6            ;
3081  020754  026364                    $DIV                    ;
3082  020756  023566                    $ADD                    ;
3083  020760  025244                    $MUL                    ;
3084  020762  023510  002506                    $PUSH   ,OP5            ;
3085  020766  025244                    $MUL                    ;
3086  020770  023540  002416                    $POPX   ,ANS2           ;
3087
3088  020774  170127  040000                    LDFPS   #040000         ;NO CHECKS
3089  021000  172437  002416                    LDF     ANS2,AC0        ;GET SOFTWARE ANSWER
3090  021004  013700  002400                    MOV     $FPS,R0         ;DISPLAY $FPS
3091  021010  012737  021016  001114            MOV     #,+6,$LPADR     ;RESET LOOP ADDRESS
3092
3093                                    ;;*************************************************************
3094
3095  021016  170127  000440                    LDFPS   #000440         ;INITIAL FPS
3096  021022  172533  002446                    LDF     OP1,AC1         ;AC1 <= OP1
3097  021026  173137  002456                    SUBF    OP2,AC1         ;AC1 <= OP1-OP2
3098  021032  172637  002476                    LDF     OP4,AC2         ;AC2 <= OP4
3099  021036  174637  002516                    DIVF    OP6,AC2         ;AC2 <= OP4/OP6
3100  021042  172237  002466                    ADDF    OP3,AC2         ;AC2 <= OP3+OP4/OP6
3101  021046  171102                    MULF    AC2,AC1         ;AC1 <= (OP1-OP2)*(OP3+OP4/OP6)
3102  021050  171137  002506                    MULF    OP5,AC1         ;AC1 <= (OP1-OP2)*(OP3+OP4/OP6)*OP5
3103
3104  021054  170237  002362                    STFPS   FPS             ;STORE STATUS AFTERWARD
3105  021060  023737  002362  002400            CMP     FPS,$FPS        ;CHECK STATUS
3106  021066  001403                    BEQ     ETST1           ;BRANCH IF OK
3107  021070  174137  002406                    STF     AC1,ANS1        ;SAVE FPU ANSWER
```



```
3108  021074  104033                    ERROR   33              ;FPS ERROR
3109
3110  021076  173401                    ETST1:  CMPF    AC1,AC0         ;ANSWER OK ? (FPU:SOFTWARE)
3111  021100  170000                    CFCC                    ;COPY CC+S
3112  021102  001436                    BEQ     EEND1           ;ANSWER CHECKS
3113
3114                                    ;COMPENSATE IN LOB FOR INACCURACIES
3115  021104  174137  002406                    STF     AC1,ANS1        ;FPU ANSWER
3116
3117  021110  163737  002420  002410            SUB     ANS2+2,ANS1+2   ;GET (SOFT-ANS) - (FPU-ANS)
3118  021116  005637  002406                    SBC     ANS1+0          ;
3119  021122  163737  002416  002406            SUB     ANS2+0,ANS1+0   ;
3120  021130  100011                    BPL     10$             ;ALWAYS MAKE +
3121  021132  005137  002410                    COM     ANS1+2          ;
3122  021136  005137  002406                    COM     ANS1+0          ;
3123  021142  062737  000001  002410            ADD     #1,ANS1+2       ;
3124  021150  005537  002406                    ADC     ANS1+0          ;
3125
3126  021154  005737  002406            10$:    TST     ANS1+0          ;
3127  021160  001034                    BNE     EERR1           ;IF NONZERO IN 16 MOR, SIGN/EXP/FRAC DIFFERS
3128
3129  021162  023727  002410  000006            CMP     ANS1+2,#6       ;ALLOW +/- 6 IN LSB OF FRAC
3130  021170  003403                    BLE     EEND1           ;BR IF OK
3131
3132  021172  174137  002406            EERR1:  STF     AC1,ANS1        ;FPU ANSWER
3133  021176  104035                    ERROR   35              ;ANSWERS DON'T CHECK
3134
3135  021200  005037  002362            EEND1:  CLR     FPS             ;CLEAR BUFFER
3136
3137
3138
3139
3140                                    ;;*************************************************************
3141                                    ;*TEST 46        ADDD, SUBD, MULD, DIVD EXERCISER
3142                                    ;;*************************************************************
3143  021204  000001                    TST46:  SCOPE
3144                                    ;*UNDERFLOW, OVERFLOW INTERRUPTS OFF; ROUND MODE
3145
3146  021206  012737  000600  002400            MOV     #600,$FPS       ;SOFTWARE STATUS
3147  021214  005037  002402                    CLR     $FEC            ;CLEAR STATUS
3148  021220  005037  002404                    CLR     $FEA            ;
3149  021224  005037  002362                    CLR     FPS             ;
3150  021230  005037  002364                    CLR     FEC             ;
3151  021234  005037  002366                    CLR     FEA             ;
3152
3153  021240  004737  023332                    JSR     PC,RANDL4       ;GET 6 DOUBLE FLOAT RANDOM NUMBERS
3154  021244  002446  002476                    .WORD   OP1,OP4         ;
3155  021250  004737  023332                    JSR     PC,RANDL4       ;
3156  021254  002466  002456                    .WORD   OP3,OP2         ;
3157  021260  004737  023332            15:     JSR     PC,RANDL4       ;
3158  021264  002506  002516                    .WORD   OP5,OP6         ;
3159  021270  032737  077600  002506            BIT     #077600,OP5     ;LET'S NEVER DIVIDE BY ZERO
3160  021276  001770                    BEQ     15              ;OP5 IS ZERO, TRY AGAIN
3161  021300  032737  077600  002516            BIT     #077600,OP6     ;
3162  021306  001764                    BEQ     15              ;OP6 IS ZERO, TRY AGAIN
3163
```

```
 3164  021310  004437  023506                JSR     R4,$POLSH       ;ENTER POLISH MODE TO CALCULATE:
 3165  021314  023510  002446                $PUSH   ,OP1            ;
 3166  021320  023510  002466                $PUSH   ,OP3            ;          /OP1+OP3\   /OP2-OP4\
 3167  021324  023566                        $ADD                    ;  ANS2 = I-------I * I-------I
 3168  021326  023510  002506                $PUSH   ,OP5            ;          \  OP5  /   \  OP6  /
 3169  021332  026364                        $DIV                    ;
 3170  021334  023510  002456                $PUSH   ,OP2            ;
 3171  021340  023510  002476                $PUSH   ,OP4            ;
 3172  021344  023562                        $SUB                    ;
 3173  021346  023510  002516                $PUSH   ,OP6            ;
 3174  021352  026364                        $DIV                    ;
 3175  021354  025244                        $MUL                    ;
 3176  021356  023540  002416                $FOPX   ,ANS2           ;
 3177
 3178  021362  170127  040200                LDFPS   #04#200         ;NO CHECKS
 3179  021366  172437  002416                LDD     ANS2,AC0        ;GET SOFTWARE ANSWER
 3180  021372  013700  002400                MOV     $FPS,R0         ;DISPLAY $FPS
 3181  021376  012737  021404  001110        MOV     #,+6,$LPADR     ;RESET LOOP ADDRESS
 3182
 3183                          ;;*************************************************************
 3184
 3185  021404  170127  000600                LDFPS   #000600         ;INITIAL FPS
 3186  021410  172537  002446                LDD     OP1,AC1         ;AC1 <- OP1
 3187  021414  172137  002466                ADDD    OP3,AC1         ;AC1 <- OP1+OP3
 3188  021420  174537  002506                DIVD    OP5,AC1         ;AC1 <- (OP1+OP3)/OP5
 3189  021424  172637  002456                LDD     OP2,AC2         ;AC2 <- OP2
 3190  021430  173237  002476                SUBD    OP4,AC2         ;AC2 <- OP2-OP4
 3191  021434  174637  002516                DIVD    OP6,AC2         ;AC2 <- (OP2-OP4)/OP6
 3192  021440  171102                        MULD    AC2,AC1         ;AC1 <- (OP1+OP3)/OP5*(OP2-OP4)/OP6
 3193
 3194  021442  170237  002362                STFPS   FPS             ;STORE STATUS AFTERWARD
 3195  021446  023737  002362  002400        CMP     FPS,$FPS        ;CHECK STATUS
 3196  021454  001403                        BEQ     ETST2           ;BRANCH IF OK
 3197  021456  174137  002406                STD     AC1,ANS1        ;SAVE FPU ANSWER
 3198  021462  104034                        ERROR   34              ;FPS ERROR
 3199
 3200  021464  173401                ETST2:  CMPD    AC1,AC0         ;ANSWER OK ? (FPU:SOFTWARE)
 3201  021466  170000                        CFCC                    ;COPY CC-S
 3202  021470  001474                        BEQ     EEND2           ;ANSWER CHECKS
 3203
 3204                                  ;COMPENSATE IN LOB FOR INACCURACIES
 3205  021472  174137  002406                STD     AC1,ANS1        ;FPU ANSWER
 3206
 3207  021476  163737  002424  002414        SUB     ANS2+6,ANS1+6   ;GET (SOFT-ANS) - (FPU-ANS)
 3208  021504  005637  002412                SBC     ANS1+4          ;
 3209  021510  005637  002410                SBC     ANS1+2          ;
 3210  021514  005637  002406                SBC     ANS1+0          ;
 3211  021520  163737  002422  002412        SUB     ANS2+4,ANS1+4   ;
 3212  021526  005637  002410                SBC     ANS1+2          ;
 3213  021532  005637  002406                SBC     ANS1+0          ;
 3214  021536  163737  002420  002410        SUB     ANS2+2,ANS1+2   ;
 3215  021544  005637  002406                SBC     ANS1+0          ;
 3216  021550  163737  002416  002406        SUB     ANS2+0,ANS1+0   ;
 3217  021556  100021                        BPL     10$             ;ALWAYS MAKE +
 3218  021560  005137  002414                COM     ANS1+6          ;
 3219  021564  005137  002412                COM     ANS1+4          ;
```



```
 3220  021570  005137  002410                COM     ANS1+2          ;
 3221  021574  005137  002406                COM     ANS1+0          ;
 3222  021600  062737  000001  002414        ADD     #1,ANS1+6       ;
 3223  021606  005537  002412                ADC     ANS1+4          ;
 3224  021612  005537  002410                ADC     ANS1+2          ;
 3225  021616  005537  002406                ADC     ANS1+0          ;
 3226
 3227  021622  005737  002406        10$:    TST     ANS1+0          ;
 3228  021626  001012                        BNE     EERR2           ;IF NONZERO IN 16 MOB, SIGN/EXP/FRAC DIFFERS
 3229  021630  005737  002410                TST     ANS1+2          ;
 3230  021634  001007                        BNE     EERR2           ;IF NONZERO IN 16 M-MOB, FRAC-B DIFFERS
 3231  021636  005737  002412                TST     ANS1+4          ;
 3232  021642  001004                        BNE     EERR2           ;IF NONZERO IN 16 L-MOB, FRAC-C DIFFERS
 3233
 3234  021644  023727  002414  000005        CMP     ANS1+6,#5       ;ALLOW +/- 5 IN LSB OF FRAC
 3235  021652  003403                        BLE     EEND2           ;BR IF OK
 3236
 3237  021654  174137  002406        EERR2:  STD     AC1,ANS1        ;FPU ANSWER
 3238  021660  104036                        ERROR   36              ;ANSWERS DON'T CHECK
 3239
 3240  021662  005037  002362        EEND2:  CLR     FPS             ;CLEAR BUFFER
```

```
                                ;;*****************************************************************
                                .SBTTL  SUB PASS END CONTROL
                                
 021666  000004                         SCOPE           ;CHECK FOR TEST ITERATIONS HERE
 021670  005037  001166                 CLR     $TIMES          ;DONT ITERATE THIS "TEST"
 021674  005037  001104                 CLR     $ERFLG          ;NO ERRORS HERE
 021700  005037  001102                 CLR     $TSTNM          ;ZAP TEST # WHEN DONE WITH A PASS
                                
                                ;IF TEST ONLY EITHER HFP OR WFP, ENTER "EOP" ROUTINE DIRECTLY
                                
                                ; IF IN ALTERNATE HFP/WFP MODE,
                                ;   COMPLEMENT FLAG<5>, HFP ENABLE BIT,
                                ;   ENTER EOP ROUTINE ONLY IF ABOUT TO TEST HFP NEXT,
                                ;       TESTING SEQUENCE IS:  PASS#1  HFP SUB-PASS
                                ;                             PASS#1  WFP SUB-PASS
                                ;                             PASS#2  HFP SUB-PASS
                                ;                               ...
                                
 021704  176600  000022                 MFPS    ,RWHAMI         ;GET WHAMI INTO R0
 021710  132700  000020                 BIT     #BIT04,R0       ;1=HFP PRESENT, 0=NONE
 021714  001423                         BEQ     $EOP            ;EXIT IF NONE
                                
 021716  032777  000002  157220         BIT     #SW01,@SWR      ;1=HFP OR WFP TEST ONLY
 021724  001017                         BNE     $EOP            ;0=ALTERNATE HFP AND WFP TESTS
                                
 021726  012701  010000                 MOV     #BIT12,R1       ;HFP PRESENT, AND IN ALTERNATE MODE;
 021732  176600  000144                 MFPS    ,RFLAG          ;SO READ FLAGS,
 021736  130100                         BIT     R1,R0           ;COMPLEMENT FLAG<5>=BIT12=HFP ENABLE FLAG
 021740  001402                         BEQ     1$              ;
 021742  040100                         BIC     R1,R0           ;CLEAR BIT 12
 021744  000401                         BR      2$              ;
 021746  050100                 1$:     BIS     R1,R0           ;SET BIT 12
 021750  176600  000344         2$:     MTPS    ,WFLAG          ;REWRITE FLAGS
                                
 021754  130100                         BIT     R1,R0           ;HFP OR WFP NEXT ?
 021756  001002                         BNE     $EOP            ;IF HFP AGAIN, START NEW PASS
 021760  000137  003516                 JMP     @#SUBPAS        ;IF WFP, NEXT SUBPASS
                                
                                
                                ;;*****************************************************************
                                
                                .SBTTL  END OF PASS ROUTINE (MODIFIED SYSMAC)
                                
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*LOOP FOR 256. SUBPASSES TO MAKE A PASS
                                ;*IF SW<10>=0, DING BELL ON PASS END
                                ;*IF SW<12>=0, TYPE STATISTICS ON PASS END
                                ;*IF THERE'S A MONITOR, GO TO IT
                                ;*  ELSE JUMP TO NEWPAS
                                
 021764                         $EOP:
 021764  005037  001104                 CLR     $ERFLG          ;ZERO ERROR FLAG
 021770  005037  001102                 CLR     $TSTNM          ;ZERO TEST NUMBER
 021774  005037  001166                 CLR     $TIMES          ;ZERO NUMBER OF ITERATIONS
                                
 022000  005327                         DEC     (PC)+           ;SUBPASS LOOP ?
```



```
 022002  000400                 10$:    .WORD   256.            ;USE 256. SUBPASSES PER LOOP
 022004  003402                         BLE     11$             ;NO, GO BUMP PASS COUNTER
 022006  000137  003516                 JMP     @#SUBPAS        ;NEXT SUB PASS
 022012  012737                 11$:    MOV     (PC)+,@(PC)+    ;RESTORE COUNTER
 022014  000400                         .WORD   256.            ;
 022016  022002                         .WORD   10$             ;
                                
 022020  005237  001210                 INC     $PASS           ;INCREMENT PASS COUNT,
 022024  042737  100000  001210         BIC     #100000,$PASS   ;  BUT NEVER LET IN GO NEGATIVE
 022032  005327                         DEC     (PC)+           ;PASS LOOP ?
 022034  000001                 $EOPCT: .WORD   1               ;FALL THRU
 022036  003027                         BGT     $DOAGN          ;YES
 022040  012737                         MOV     (PC)+,@(PC)+    ;RESTORE COUNTER
 022042  000001                 $ENDCT: .WORD   1               ;
 022044  022034                         .WORD   $EOPCT          ;
                                
 022046  032777  002000  157070         BIT     #SW10,@SWR      ;BELL ON PASS END ?
 022054  001002                         BNE     1$              ;NO
 022056  104401  001172                 TYPE    ,$BELL          ;YES
                                
 022062  032777  010000  157054  1$:    BIT     #SW12,@SWR      ;INHIBIT MESSAGE ?
 022070  001002                         BNE     $GET42          ;YES
 022072  004737  022122                 JSR     PC,STATS        ;TYPE STATISTICS
                                
 022076  013700  000042         $GET42: MOV     @#42,R0         ;GET MONITOR ADDRESS
 022102  001405                         BEQ     $DOAGN          ;NO MONITOR
 022104  000005                         RESET                   ;CLEAR WORLD
 022106  004710                 $ENDAD: JSR     PC,(R0)         ;GO TO MONITOR
 022110  000240                         NOP                     ;
 022112  000240                         NOP                     ;RESERVED FOR ACT11
 022114  000240                         NOP                     ;
                                
 022116  000137  003450         $DOAGN: JMP     @#NEWPAS        ;RETURN
```

```
                                        .SBTTL  STATISTICS TYPEOUT SUBROUTINE

                                        ;*THIS ROUTINE TYPES OUT A NICELY FORMATTED REPORT
                                        ;*CONTAINING STATISTICS ON THE NUMBER OF OPERANDS
                                        ;*USED IN DIFFERENT SELECT CASES FOR THE $ADD, $SUB,
                                        ;*$MUL, AND $DIV ROUTINES.  THE DATA VALUES ARE TAKEN
                                        ;*FROM THE COUNTERS ADDC?, MULC?, AND DIVC?.
                                        ;*
                                        ;*CALLED BY:    JSR     PC,STATS
                                        ;*
022122  010046                  STATS:  MOV     R0,-(SP)        ;SAVE REGISTERS

022124  012700  022230                  MOV     #SVEC1,R0       ;FIRST DATA LINE ADDR VECTOR
022130  001404                          BEQ     1$              ;NONE IF ZERO
022132  104401  022317                  TYPE    ,SHDR1          ;HEADER
022136  004737  022202                  JSR     PC,10$          ;DATA LINE
022142  012700  022254          1$:     MOV     #SVEC2,R0       ;SECOND DATA LINE ADDR VECTOR
022146  001404                          BEQ     2$              ;NONE IF ZERO
022150  104401  022664                  TYPE    ,SHDR2          ;HEADER
022154  004737  022202                  JSR     PC,10$          ;DATA LINE
022160  012700  022272          2$:     MOV     #SVEC3,R0       ;THIRD DATA LINE ADDR VECTOR
022164  001404                          BEQ     3$              ;NONE IF ZERO
022166  104401  023055                  TYPE    ,SHDR3          ;HEADER
022172  004737  022202                  JSR     PC,10$          ;DATA LINE

022176  012600                  3$:     MOV     (SP)+,R0        ;RESTORE REGISTERS
022200  000207                          RTS     PC              ;EXIT

                                        ;*INTERNAL SUBR FOR DATA LINE TYPEOUT
022202  013046                  10$:    MOV     @(R0)+,-(SP)    ;MOVE NUMBER ON STACK, BUMP TO NEXT
022204  104403                          TYPOS                   ;TYPE OCTAL
022206     006                          .BYTE   6               ;MAX 6 DIGITS
022207     000                          .BYTE   0               ;SUPPRESS LEADING ZEROS
022210  005710                          TST     (R0)            ;BUMPED TO LAST VECTOR ?
022212  001403                          BEQ     11$             ;YES, ONTO NEXT LINE
022214  104401  022312                  TYPE    ,SHT            ;TYPE A <HT>
022220  000770                          BR      10$             ;CONTINUE WITH THIS LINE
022222  104401  022314          11$:    TYPE    ,SCRLF          ;TYPE A <CR><LF>
022226  000207                          RTS     PC              ;DONE

                                        ;*DATA VECTORS:
022230  002526  002540  002542  SVEC1:  .WORD   ADDC0,ADDC5,ADDC6,ADDC7,ADDC8,ADDC1,ADDC2,ADDC3,ADDC4,0
022236  002544  002546  002530
022244  002532  002534  002536
022252  000000
022254  002550  002554  002556  SVEC2:  .WORD   MULC0,MULC2,MULC3,MULC4,MULC5,MULC1,0
022262  002560  002562  002552
022270  000000
022272  002564  002572  002574  SVEC3:  .WORD   DIVC0,DIVC3,DIVC4,DIVC5,DIVC6,DIVC1,DIVC2,0
022300  002576  002600  002566
022306  002570  000000

                                        ;*HEADERS, ETC:
022312  000011                  SHT:    .ASCIZ  <11>            ;<HT>
022314  005015     000          SCRLF:  .ASCIZ  <15><12>        ;<CR><LF>
```



```
022317     015  005012          SHDR1:  .ASCII  <15><12><12>
022322  025052  020052  052123          .ASCII  "*** STATISTICS ***"<15><12><12>
022330  052101  051511  044524
022336  051503  025040  025052
022344  005015     012
022347     050  047516  042524          .ASCII  "(NOTE - ALL NUMBERS ARE UNSIGNED OCTAL INTEGERS)"<15><12><12>
022354  026440  040440  046114
022362  047040  046525  042502
022370  051522  040440  042522
022376  052440  051516  043511
022404  042516  020104  041517
022412  040524  020114  047111
022420  042524  042507  051522
022426  006451  005012
022432  042101  027504  052523          .ASCII  "ADD/SUB INSTRUCTIONS:"<15><12>
022440  020102  047111  052123
022446  052522  052103  047511
022454  051516  006472     012
022461     124  052117  046101          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---  OP A=0  OP A=0  OP A=0     NO"<15><12>

022466  026411  026455  053117
022474  051105  046106  053517
022502  026455  026455  026455
022510  047125  042504  043122
022516  047514  026527  026455
022524  047411  020120  036501
022532  004460  050117  040440
022540  030043  047411  020120
022546  036501  004460  020040
022554  047040  006517     012
022561     116  046525  042502          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE  OP B=0  OP B=0  OP B=0   SHIFT"<15><12>

022566  004522  041524  040524
022574  020114  027527  047105
022602  041101  042514  052011
022610  052117  046101  053440
022616  042457  040516  046102
022624  004505  050117  041040
022632  030043  047411  020120
022640  036502  004460  050117
022646  041040  030075  020011
022654  044123  043111  006524
022662  000012
022664  005015  052515  052114  SHDR2:  .ASCII  <15><12>"MULTIPLY INSTRUCTIONS:"<15><12>
022672  050111  054514  044440
022700  051516  051124  041525
022706  044524  047117  035123
022714  005015
022716  047524  040524  004514          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---  A=0 AND/OR"<15><12>
022724  026455  047455  042526
022732  043122  047514  026527
022740  026455  026411  052455
022746  042116  051105  046106
022754  053517  026455  004455
022762  036501  020060  047101
022770  027504  051117  005015
```

```
3440  023776  052516  041115  051105          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE      B=0"<15><12>
3441  024004  052011  052117  046101
3442  024012  053440  042457  040516
3443  024020  046102  004505  047524
3444  024026  040524  020114  027527
3445  024034  047105  041101  042514
3446  024042  020011  020040  041040
3447  024050  030075  005015      000
3448  024055      015  042012  053111  SND23:  .ASCII  <15><12>"DIVIDE INSTRUCTIONS!"<15><12>
3449  024062  042111  020105  047111
3450  024070  052123  052522  052103
3451  024076  047511  051516  006472
3452  024104      012
3453  024105      124  052117  046101          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---   NUMER   DENOM"<15><12>
3454  024112  026411  026455  053117
3455  024120  051105  046106  053517
3456  024126  026455  004455  026455
3457  024134  047125  042504  043122
3458  024142  047514  026527  026455
3459  024150  020011  052516  042515
3460  024156  004522  042040  047105
3461  024164  046517  005015
3462  024170  052516  041115  051105          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE      A=0     B=0"<15><12>
3463  024176  052011  052117  046101
3464  024204  053440  042457  040516
3465  024212  046102  004505  047524
3466  024220  040524  020114  027527
3467  024226  047105  041101  042514
3468  024234  020011  020040  036501
3469  024242  004460  020040  041040
3470  024250  030075  005015      000
3471          024256                          .EVEN           ;BACK TO AN EVEN BOUNDARY
```



```
3472                                          .SBTTL  FPP TRAP CATCHER
3473
3474  024256  012637  002370          FPPILT: MOV     (SP)+,FPPOPC    ;POP OLD PC FOR DISPLAY
3475  024262  012637  002372                  MOV     (SP)+,FPPOPS    ;POP OLD PS FOR DISPLAY
3476  024266  170237  002362                  STFPS   FPS             ;GET FPS
3477  024272  170337  002364                  STST    FEC             ;GET FEC/FEA
3478  024276  005737  002362                  TST     FPS             ;TEST ERROR BIT
3479  024302  100005                          BPL     1$              ;OFF - NO ERROR BIT SET, BUT TRAPPED
3480                                                                  ;ON  - IT SHOULD BE ON A TRAP
3481  024304  032737  040000  002362          BIT     #040000,FPS     ;TEST INTERRUPT ENABLE BIT
3482  024312  001001                          BNE     1$              ;ON  - INTR DISABLED, BUT TRAPPED
3483                                                                  ;OFF - ABLE TO INTR, SO IGNORE IT,
3484  024314  000401                          BR      2$              ;       AND SKIP THE ERROR
3485  024316  104015                  1$:     ERROR   15              ;SIGNAL UNEXPECTED FPP TRAP
3486  024320  013746  002372          2$:     MOV     FPPOPS,-(SP)    ;PUSH PSW
3487  024324  013746  002370                  MOV     FPPOPC,-(SP)    ;PUSH PC
3488  024330  000002                          RTI                     ;CONTINUE,  RECOVER AT LAST TRAP ONLY
3489
```

```
                                        .SBTTL  RANDOM NUMBER GENERATOR

                                        ;*CALLED BY     JSR  PC,RANDL4  -  FOR DOUBLE FLOAT NUMBERS
                                        ;*              .WORD   N1,N2   -   IN LOCATIONS N1 AND N2
                                        ;*
                                        ;*              JSR  PC,RANDL2  -  FOR SINGLE FLOAT NUMBERS
                                        ;*              .WORD   N1,N2   -   IN LOCATIONS N1 AND N2
                                        ;*
023332  010546              RANDL4: MOV     R5,-(SP)        ;SAVE R5
023334  012705  000004              MOV     #4,R5           ;4 WORDS AT EACH
023340  000403                      BR      RAND            ;
023342  010546              RANDL2: MOV     R5,-(SP)        ;SAVE R5
023344  012705  000002              MOV     #2,R5           ;2 WORDS AT EACH
023350  010446              RAND:   MOV     R4,-(SP)        ;SAVE REGISTERS
023352  010346                      MOV     R3,-(SP)        ;
023354  010246                      MOV     R2,-(SP)        ;
023356  010146                      MOV     R1,-(SP)        ;
023360  010046                      MOV     R0,-(SP)        ;
023362  005046                      CLR     -(SP)           ;EXTRA REGISTER
023364  016600  000016              MOV     16(SP),R0       ;GET PC FOR RETURN
023370  012003                      MOV     (R0)+,R3        ;FIRST NUMBER DEST PTR
023372  012004                      MOV     (R0)+,R4        ;SECOND NUMBER DEST PTR
023374  010066  000016              MOV     R0,16(SP)       ;STORE NEW RETURN ADDRESS
023400  011300                      MOV     (R3),R0         ;R0 INITIAL NUMBER
023402  011401                      MOV     (R4),R1         ;R1 INITIAL NUMBER
023404  012702  000007      1$:     MOV     #7,R2           ;SHIFT COUNT
023410  005016                      CLR     (SP)            ;CLEAR LOB
023412  006300              2$:     ASL     R0              ;SHIFT R0 LEFT
023414  006101                      ROL     R1              ;  AND ROTATE CARRY INTO R1
023416  006116                      ROL     (SP)            ;  AND ROTATE CARRY INTO EXT
023420  077204                      SOB     R2,2$           ;7 SHIFTS
023422  061316                      ADD     (R3),(SP)       ;ADD # TO MAKE * 129
023424  005501                      ADC     R1              ;PROPOGATE CARRY
023426  061401                      ADD     (R4),R1         ;ADD # TO MAKE * 129
023430  005516                      ADC     (SP)            ;PROPOGATE CARRY
023432  062700  001057              ADD     #1057,R0        ;ADD LOW CONSTANT
023436  005501                      ADC     R1              ;PROPOGATE CARRY
023440  005516                      ADC     (SP)            ;PROPOGATE CARRY
023442  062701  047401              ADD     #47401,R1       ;ADD HIGH CONSTANT
023446  005516                      ADC     (SP)            ;PROPOGATE CARRY
023450  062716  000006              ADD     #6,(SP)         ;ADD HIGHEST CONSTANT
023454  061600                      ADD     (SP),R0         ;REPRIME R0 WITH HIGHEST DIGIT
023456  005501                      ADC     R1              ;PROPOGATE CARRY
023460  010023                      MOV     R0,(R3)+        ;SAVE R0
023462  010124                      MOV     R1,(R4)+        ;SAVE R1
023464  077531                      SOB     R5,1$           ;LOOP FOR REQ'D NUMBER OF WORDS
023466  005726                      TST     (SP)+           ;POP TEMP REG
023470  012600                      MOV     (SP)+,R0        ;
023472  012601                      MOV     (SP)+,R1        ;RESTORE REGISTERS
023474  012602                      MOV     (SP)+,R2        ;
023476  012603                      MOV     (SP)+,R3        ;
023500  012604                      MOV     (SP)+,R4        ;
023502  012605                      MOV     (SP)+,R5        ;
023504  000207              RTS     PC                      ;RETURN
```



```
                                        .SBTTL  POLISH EXPRESSION ROUTINES

                                        ;*"POLISH EXPRESSION" ALGEBRA IS A STACK ORIENTED PROCEDURE FOR
                                        ;*THE EVALUATION OF ALGEBRAIC EXPRESSIONS.  BY THIS, WE MEAN
                                        ;*THAT THE STACK (IN OUR CASE, WE WILL BE USING THE STANDARD
                                        ;*SYSTEM STACK USING R6) IS USED FOR THE STORAGE, ON A LAST IN-
                                        ;*FIRST OUT BASIS, OF ALL OPERANDS, AND RESULTS.  ALL THE
                                        ;*ARITHMETIC ROUTINES (SPECIFICALLY, OUR $ADD, $SUB, $MUL, AND
                                        ;*$DIV ROUTINES) EXPECT THEIR TWO OPERANDS TO BE THE TOP TWO
                                        ;*ELEMENTS ON THE STACK, AND THEY LEAVE THEIR RESULT AS THE
                                        ;*TOP ELEMENT ON THE STACK, REMOVING (POPPING) THE INITIAL
                                        ;*OPERANDS IN THE PROCESS.  OTHER ROUTINES ARE PRESENT FOR
                                        ;*ADDING/REMOVING ELEMENTS TO/FROM THE STACK - $POPX, TO TAKE
                                        ;*THE TOP ELEMENT OFF, AND $PUSH, TO PUT A NEW ELEMENT ON THE
                                        ;*TOP.  IT IS IMPORTANT TO NOTE THAT OPERATORS WILL AT MOST
                                        ;*REFERENCE THE TOP TWO ELEMENTS ON THE STACK; THE OTHERS ARE
                                        ;*INACCESSIBLE UNTIL OUTER ELEMENTS ARE OPERATED UPON.  FOR
                                        ;*EXAMPLE, THE EXPRESSION:
                                        ;*
                                        ;*      E = (A + B) * (C - D)
                                        ;*
                                        ;*COULD BE EVALUATED BY THE POLISH EXPRESSION:
                                        ;*
                                        ;*      $PUSH   A       ;OPERAND A ONTO STACK
                                        ;*      $PUSH   B       ;OPERAND B ONTO STACK
                                        ;*      $ADD            ;FORM A+B, SAVE FOR LATER
                                        ;*      $PUSH   C       ;OPERAND C ONTO STACK
                                        ;*      $PUSH   D       ;OPERAND D ONTO STACK
                                        ;*      $SUB            ;FORM C-D ON TOP
                                        ;*                      ;NOTE - THE TOP TWO OPERANDS ARE NOW:
                                        ;*                      ;    (A+B) AND (C-D)
                                        ;*      $MUL            ;FORM (A+B) * (C-D) ON TOP
                                        ;*      $POPX   E       ;POP RESULT FROM STACK INTO E
                                        ;*
                                        ;*NOTE THAT OTHER POLISH EXPRESSIONS ARE POSSIBLE FOR COMPUTING
                                        ;*THIS EXAMPLE, IN GENERAL THERE IS MORE THAN ONE WAY TO
                                        ;*CALCULATE A GIVEN EXPRESSION.
                                        ;*

                                ;*THIS ROUTINE ENTERS US INTO POLISH MODE
023506  000134              $POLSH: JMP     @(R4)+          ;ENTER POLISH MODE

                                ;*PUSH OPERAND ON STACK FROM LOCATION SPECIFIED
                                ;*IN CONTENTS OF NEXT WORD AFTER $PUSH CALL
                                ;*2/4 WORDS DEPENDING UPON F/D MODE
023510  012400              $PUSH:  MOV     (R4)+,R0        ;GET PTR TO SOURCE
023512  105737  002400              TSTB    @FPS            ;
023516  100004                      BPL     1$              ;
023520  016046  000006              MOV     6(R0),-(SP)     ;PUSH DOUBLE ON STACK
023524  016046  000004              MOV     4(R0),-(SP)     ;
023530  016046  000002      1$:     MOV     2(R0),-(SP)     ;PUSH FLOAT ON STACK
023534  011046                      MOV     (R0),-(SP)      ;
023536  000134                      JMP     @(R4)+          ;

                                ;*POP OPERAND FROM STACK INTO LOCATION SPECIFIED
                                ;*IN CONTENTS OF NEXT WORD AFTER $POPX CALL
```

```
 3601                                   ;*2/4 WORDS DEPENDING UPON F/D MODE
 3602  023540  012400                  $POPX:  MOV     (R4)+,R0        ;GET PTR TO DESTINATION
 3603  023542  012620                          MOV     (SP)+,(R0)+     ;POP FLOAT FROM STACK
 3604  023544  012620                          MOV     (SP)+,(R0)+     ;
 3605  023546  105737  002400                  TSTB    $FPS            ;
 3606  023552  100002                          BPL     1$              ;
 3607  023554  012620                          MOV     (SP)+,(R0)+     ;POP DOUBLE FROM STACK
 3608  023556  012620                          MOV     (SP)+,(R0)+     ;
 3609  023560  000204                  1$:     RTS     R4              ;EXIT POLISH MODE
```



```
 3610                                   .SBTTL  FLOATING POINT SOFTWARE ROUTINES
 3611
 3612                                   ;*FLOATING POINT SOFTWARE ADD/SUBTRACT ROUTINES
 3613                                   ;*
 3614                                   ;*  THIS ROUTINE TAKES THE TOP TWO ELEMENTS ON THE STACK
 3615                                   ;*    (LENGTH OF 2 OR 4 WORDS DEPENDING UPON WHETHER THE
 3616                                   ;*    F/D MODE BIT7 IN $FPS IS 0 OR 1 RESPECTIVELY) AND
 3617                                   ;*    REPLACES THEM WITH THEIR SUM/DIFFERENCE.
 3618                                   ;*
 3619                                   ;*  EXAMPLE:    $PUSH                   $PUSH
 3620                                   ;*              ADDR(OPERAND A)         ADDR(OPERAND A)
 3621                                   ;*              $PUSH                   $PUSH
 3622                                   ;*              ADDR(OPERAND B)         ADDR(OPERAND B)
 3623                                   ;*              $ADD                    $SUB
 3624                                   ;*              $POPX                   $POPX
 3625                                   ;*              ADDR(RESULT)            ADDR(RESULT)
 3626                                   ;*                RESULT=A+B              RESULT=A-B
 3627                                   ;*
 3628                                   ;*  NOTE ROUTINE IS CALLED THROUGH POLISH MODE OPERATORS.
 3629                                   ;*
 3630                                   ;*  ALL OPERATIONS ARE CARRIED OUT IN DOUBLE (LENGTH=4)
 3631                                   ;*    PRECISION MODE.  SINGLE PRECISION OPERANDS ARE
 3632                                   ;*    EXTENDED (VIA $CONV SUBROUTINE) WITH LOW-ORDER
 3633                                   ;*    ZEROS, AND LATER TRUNCATED BACK TO SINGLE PRECISION
 3634                                   ;*    FROM THE DOUBLE PRECISION RESULT.
 3635                                   ;*
 3636                                   ;*  STATUS BITS:
 3637                                   ;*    THE N, Z, V, AND C BITS OF $FPS ARE SET AS FOLLOWS:
 3638                                   ;*    N = 1  IF RESULT NEGATIVE (IE, BIT15 = 1),
 3639                                   ;*           ELSE  N = 0
 3640                                   ;*    Z = 1  IF RESULT ZERO (IE, EXPONENT = 000(8)),
 3641                                   ;*           ELSE  Z = 0
 3642                                   ;*    V = 1  IF ARITHMETIC OVERFLOW OCCURRED,
 3643                                   ;*           ELSE  V = 0
 3644                                   ;*    C = 0  ALWAYS
 3645                                   ;*
 3646                                   ;*  ERROR CONDITIONS:
 3647                                   ;*    IF AN ARITHMETIC OVERFLOW CONDITION OCCURS, THE RESULT
 3648                                   ;*    WILL BE SET TO ZERO IF THE OVERFLOW ENABLE BIT OF $FPS
 3649                                   ;*    (BIT09) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
 3650                                   ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE UNDERBIASED
 3651                                   ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
 3652                                   ;*    $FEC WILL BE SET TO 10(8), AND $FEA WILL BE LOADED WITH
 3653                                   ;*    THE VALUE IN LOCATION "EXPFEA".  IN EITHER INSTANCE,
 3654                                   ;*    THE V-BIT (BIT01) WILL BE SET.
 3655                                   ;*    IF AN ARITHMETIC UNDERFLOW CONDITION OCCURS, THE RESULT
 3656                                   ;*    WILL BE SET TO ZERO IF THE UNDERFLOW ENABLE BIT OF $FPS
 3657                                   ;*    (BIT10) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
 3658                                   ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE OVERBIASED
 3659                                   ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
 3660                                   ;*    $FEC WILL BE SET TO 12(8), AND $FEA WILL BE LOADED WITH
 3661                                   ;*    THE VALUE IN LOCATION "EXPFEA".
 3662                                   ;*
 3663                                   ;*  ALL REGISTERS ARE PRESERVED UPON ENTRY, AND RESTORED UPON
 3664                                   ;*    EXIT.  THE ROUTINES ARE RE-ENTRANT.
 3665                                   ;*
```

```
                                        ;*  ADAPTED FROM PDP-11 FORTRAN SOFTWARE
                                        ;*    BY DONALD NORTH, SEPTEMBER, 1975.
                                        ;*
023562 062716 100000          $SUB:  ADD     #100000,(SP)    ;CHANGE SIGN OF TOP ITEM
023566 005237 002526          $ADD:  INC     ADDC0           ;CTR:  TOTAL NUMBER OF ADD/SUB
023572 042737 000017 002400          BIC     #17,$FPS        ;CLEAR STATUS BITS N Z V C
023600 105737 002400                 TSTB    $FPS            ;TEST MODE
023604 100402                        BMI     27$             ;0=MODE
023606 004737 027622                 JSR     PC,$CONV        ;0=MODE: CONVERT 2 F OPDS TO D
023612 010046                 27$:   MOV     R0,-(SP)        ;
023614 010146                        MOV     R1,-(SP)        ;
023616 010246                        MOV     R2,-(SP)        ;SAVE ALL REGISTERS
023620 010346                        MOV     R3,-(SP)        ;
023622 010446                        MOV     R4,-(SP)        ;
023624 010546                        MOV     R5,-(SP)        ;
023626 005046                        CLR     -(SP)           ;CLEAR SIGNS
023630 005004                        CLR     R4              ;CLEAR EXPONENTS
023632 005005                        CLR     R5              ;
023634 006366 000024                 ASL     24(SP)          ;SHIFT OUT SIGN OF TOP ITEM
023640 006166 000022                 ROL     22(SP)          ;
023644 006166 000020                 ROL     20(SP)          ;
023650 006166 000016                 ROL     16(SP)          ;SHIFT A1
023654 156604 000017                 BISB    17(SP),R4       ;GET E1
023660 001011                        BNE     31$             ;JUMP IF NON ZERO
023662 005726                        TST     (SP)+           ;FLUSH SIGNS
023664 032766 077600 000024          BIT     #077600,24(SP)  ;B=0, A=0 TOO ?
023672 001456                        BEQ     32$             ;YES, BOTH ZERO
023674 005237 002532                 INC     ADDC2           ;NO, CTR:  B=0,A=0
023700 000137 024712                 JMP     3$              ;DONE
023704 106116                 31$:   ROLB    (SP)            ;GET S1
023706 006366 000034                 ASL     34(SP)          ;SHIFT OUT SIGN OF 2ND ITEM
023712 006166 000032                 ROL     32(SP)          ;
023716 006166 000030                 ROL     30(SP)          ;
023722 006166 000026                 ROL     26(SP)          ;SHIFT A2
023726 156605 000027                 BISB    27(SP),R5       ;GET E2
023732 001042                        BNE     2$              ;JUMP IF NON ZERO
023734 106016                        RORB    (SP)            ;RECONSTRUCT A1
023736 006066 000016                 ROR     16(SP)          ;
023742 006066 000020                 ROR     20(SP)          ;
023746 006066 000022                 ROR     22(SP)          ;
023752 006066 000024                 ROR     24(SP)          ;
023756 016666 000016 000026          MOV     16(SP),26(SP)   ;FIRST ARG TO TOP OF STACK
023764 016666 000020 000030          MOV     20(SP),30(SP)   ;
023772 016666 000022 000032          MOV     22(SP),32(SP)   ;
024000 016666 000024 000034          MOV     24(SP),34(SP)   ;
024006 005726                        TST     (SP)+           ;FLUSH SIGNS
024010 032766 077600 000024          BIT     #077600,24(SP)  ;A=0, B=0 TOO ?
024016 001404                        BEQ     32$             ;YES, BOTH ZERO
024020 005237 002530                 INC     ADDC1           ;NO, CTR:  A=0, B#0
024024 000137 024712                 JMP     3$              ;DONE
024030 005237 002514          32$:   INC     ADDC3           ;CTR:  A=0, B=0
024034 000137 024672                 JMP     29$             ;DONE, TRUE ZERO RESULT
024040 106166 000001          2$:    ROLB    1(SP)           ;GET S2
024044 112766 000001 000027          MOVB    #1,27(SP)       ;INSERT NORMAL BIT
024052 112766 000001 000017          MOVB    #1,17(SP)       ;   "      "   "
```



```
024060 160405                        SUB     R4,R5           ;R5=E2-E1, R4=E1
024062 003011                        BGT     4$              ;JUMP IF E2>E1
024064 016600 000026                 MOV     26(SP),R0       ;R0=A2
024070 016601 000030                 MOV     30(SP),R1       ;R1=B2
024074 016602 000032                 MOV     32(SP),R2       ;R2=C2
024100 016603 000034                 MOV     34(SP),R3       ;R3=D2
024104 000427                        BR      5$              ;GO CHECK SIGNS
024106 060504                 4$:    ADD     R5,R4           ;R5=E2-E1, R4=E2,E2>E1
024110 016600 000016                 MOV     16(SP),R0       ;R0=A1
024114 016601 000020                 MOV     20(SP),R1       ;R1=B1
024120 016602 000022                 MOV     22(SP),R2       ;R2=C1
024124 016603 000024                 MOV     24(SP),R3       ;R3=D1
024130 016666 000026 000016          MOV     26(SP),16(SP)   ;
024136 016666 000030 000020          MOV     30(SP),20(SP)   ;
024144 016666 000032 000022          MOV     32(SP),22(SP)   ;
024152 016666 000034 000024          MOV     34(SP),24(SP)   ;
024160 000316                        SWAB    (SP)            ;EXCHANGE SIGNS
024162 005405                        NEG     R5              ;E1=E2
024164 126616 000001          5$:    CMPB    1(SP),(SP)      ;COMPARE SIGNS
024170 001412                        BEQ     6$              ;SAME, GO CHECK EXPONENT
024172 005403                        NEG     R3              ;NEGATE OPERAND
024174 005507                        ADC     R2              ;
024176 005501                        ADC     R1              ;
024200 005500                        ADC     R0              ;
024202 005102                        NEG     R2              ;
024204 005501                        ADC     R1              ;
024206 005500                        ADC     R0              ;
024210 005401                        NEG     R1              ;
024212 005500                        ADC     R0              ;
024214 005400                        NEG     R0              ;
024216 005705                 6$:    TST     R5              ;CHECK EXPONENTS
024220 001406                        BEQ     7$              ;JUMP IF E1=E2
024222 022705 177707                 CMP     #-57.,R5        ;ANY POINT IN SHIFTING?
024226 003413                        BLE     8$              ;YES
024230 016600 000016                 MOV     16(SP),R0       ;NO, ANSWER IS OPERAND
024234 016601 000020                 MOV     20(SP),R1       ;WITH LARGER EXPONENT
024240 016602 000022                 MOV     22(SP),R2       ;
024244 016603 000024                 MOV     24(SP),R3       ;
024250 005237 002536                 INC     ADDC4           ;CTR:  NO SHIFT
024254 000581                        BR      9$              ;
024256 022705 177770          8$:    CMP     #-8.,R5         ;CHECK # OF BITS TO SHIFT
024262 003437                        BLE     10$             ;JUMP IF LESS THAN 1/2 WORD
024264 005700                        TST     R0              ;
024266 006746                        SXT     -(SP)           ;EXTEND SIGN
024270 022705 177760          12$:   CMP     #-16.,R5        ;
024274 002411                        BLT     11$             ;JUMP IF LESS THAN 1 WORD
024276 010203                        MOV     R2,R3           ;SHIFT A WORD AT A TIME
024300 010102                        MOV     R1,R2           ;
024302 010001                        MOV     R0,R1           ;
024304 011600                        MOV     (SP),R0         ;USE EXTENSION
024306 062705 000020                 ADD     #16.,R5         ;ADJUST EXPONENT
024312 001366                        BNE     12$             ;TRY AGAIN
024314 005726                        TST     (SP)+           ;POP EXTENSION
024316 000427                        BR      7$              ;SHIFT DONE
024320 022705 177775          11$:   CMP     #-3,R5          ;JUMP IF LESS THAN 4 TO SHIFT
024324 003415                        BLE     13$             ;
```

```
 3778  024326  010416                        MOV     R4,(SP)         ;SAVE EXP & SHIFT COUNT
 3779  024330  010546                        MOV     R5,-(SP)        ;
 3780  024332  010104                        MOV     R1,R4           ;SAVE R1
 3781  024334  073005                        ASHC    R5,R0           ;SHIFT HIGH ORDER
 3782  024336  010205                        MOV     R2,R5           ;SAVE R2
 3783  024340  073416                        ASHC    (SP),R4         ;SHIFT IT
 3784  024342  010204                        MOV     R2,R4           ;
 3785  024344  010502                        MOV     R5,R2           ;R2 DONE
 3786  024346  010305                        MOV     R3,R5           ;SET UP LOW ORDER
 3787  024350  073426                        ASHC    (SP)+,R4        ;DO LOW ORDER
 3788  024352  010503                        MOV     R5,R3           ;
 3789  024354  012604                        MOV     (SP)+,R4        ;RESTORE EXP TO R4
 3790  024356  000407                        BR      7$
 3791  024360  005726                  13$:  TST     (SP)+           ;POP EXTENSION
 3792  024362  006200                  10$:  ASR     R0              ;SHIFT RIGHT
 3793  024364  006001                        ROR     R1              ;
 3794  024366  006002                        ROR     R2              ;
 3795  024370  006003                        ROR     R3              ;
 3796  024372  005205                        INC     R5              ;COUNT LOOP
 3797  024374  002772                        BLT     10$             ;
 3798  024376  066603  000024            7$:  ADD     24(SP),R3       ;FORM SUM
 3799  024402  005502                        ADC     R2              ;
 3800  024404  005501                        ADC     R1              ;
 3801  024406  005500                        ADC     R0              ;
 3802  024410  066602  000022                ADD     22(SP),R2       ;
 3803  024414  005501                        ADC     R1              ;
 3804  024416  005500                        ADC     R0              ;
 3805  024420  066601  000020                ADD     20(SP),R1       ;
 3806  024424  005500                        ADC     R0              ;
 3807  024426  066600  000016                ADD     16(SP),R0       ;
 3808  024432  126616  000001                CMPB    1(SP),(SP)      ;CHECK FOR UNEQUAL SIGNS
 3809  024436  001162                        BNE     14$             ;CLEAN UP SUBTRACT
 3810  024440  030027  001000                BIT     R0,#1000        ;
 3811  024444  001405                        BEQ     9$              ;JUMP IF NO NORMAL BIT OVERFLOW
 3812  024446  006200                        ASR     R0              ;
 3813  024450  006001                        ROR     R1              ;
 3814  024452  006002                        ROR     R2              ;
 3815  024454  006003                        ROR     R3              ;
 3816  024456  005204                        INC     R4              ;INCREASE EXP
 3817  024460  000304                  9$:   SWAB    R4              ;MOVE EXP LEFT
 3818  024462  001425                        BEQ     16$             ;JUMP IF NO OVERFLOW
 3819  024464  105004                        CLRB    R4              ;CLEAR OVERFLOWED BITS
 3820  024466  052737  000002  002400        BIS     #2,$FPS         ;SET V BIT ON OVERFLOW
 3821  024474  005237  002540                INC     ADDC5           ;CTR:  OVERFLOW
 3822  024500  032737  001000  002400        BIT     #1000,$FPS      ;OVERFLOW ENABLED ?
 3823  024506  001464                        BEQ     34$             ;NO, ZERO RESULT
 3824  024510  005237  002542                INC     ADDC6           ;CTR:  OVERFLOW, ENABLED
 3825  024514  052737  100000  002400        BIS     #100000,$FPS    ;YES, SET ERROR BIT
 3826  024522  012737  000010  002402        MOV     #10,$FEC        ;SET $FEC
 3827  024530  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
 3828  024536  150004                  16$:  BISB    R0,R4           ;INSERT HIGH ORDER FRACTION
 3829  024540  006026                        ROR     (SP)+           ;INSERT SIGN
 3830  024542  000004                        ROR     R4              ;
 3831  024544  006001                        ROR     R1              ;
 3832  024546  006002                        ROR     R2              ;
 3833  024550  006003                        ROR     R3              ;
```



```
 3834  024552  005503                        ADC     R3              ;
 3835  024554  005502                        ADC     R2              ;
 3836  024556  005501                        ADC     R1              ;
 3837  024560  005504                        ADC     R4              ;
 3838  024562  103401                        BCS     20$             ;OVERFLOW ON ROUND ?
 3839  024564  102024                        BVC     30$             ;
 3840  024566  052737  000002  002400  2$:1  BIS     #2,$FPS         ;YES - SET V BIT
 3841  024574  005237  002540                INC     ADDC5           ;CTR:  OVERFLOW
 3842  024600  032737  001000  002400        BIT     #1000,$FPS      ;OVERFLOW ENABLED ?
 3843  024606  001431                        BEQ     29$             ;NO, ZERO RESULT
 3844  024610  005237  002542                INC     ADDC6           ;CTR:  OVERFLOW, ENABLED
 3845  024614  052737  100000  002400        BIS     #100000,$FPS    ;SET ERROR BIT
 3846  024622  012737  000010  002402        MOV     #10,$FEC        ;SET $FEC
 3847  024630  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
 3848  024636  010466  000024            30$:  MOV     R4,24(SP)       ;STORE EXP AND SIGN
 3849  024642  010166  000026                  MOV     R1,26(SP)       ;INSERT LOW ORDER FRACTION
 3850  024646  010266  000030                  MOV     R2,30(SP)       ;
 3851  024652  010366  000032                  MOV     R3,32(SP)       ;
 3852  024656  000415                        BR      3$              ;
 3853  024660  005726                  34$:  TST     (SP)+           ;FLUSH SIGN
 3854  024662  000443                        BR      29$             ;AND ZERO RESULT
 3855  024664  005237  002544            33$:  INC     ADDC7           ;CTR:  UNDERFLOW
 3856  024670  005726                        TST     (SP)+           ;FLUSH SIGN
 3857  024672  005066  000024            29$:  CLR     24(SP)          ;ZERO RESULT
 3858  024676  005066  000026                  CLR     26(SP)          ;
 3859  024702  005066  000030                  CLR     30(SP)          ;
 3860  024706  005066  000032                  CLR     32(SP)          ;
 3861  024712  032766  077600  000024  3$:1  BIT     #077600,24(SP)  ;SET Z BIT IF EXPONENT ZERO
 3862  024720  001003                        BNE     17$             ;
 3863  024722  052737  000004  002400        BIS     #4,$FPS         ;
 3864  024730  005766  000024            17$:  TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
 3865  024734  100003                        BPL     18$             ;
 3866  024736  052737  000010  002400        BIS     #10,$FPS        ;
 3867  024744  012605                  18$:  MOV     (SP)+,R5        ;
 3868  024746  012604                        MOV     (SP)+,R4        ;
 3869  024750  012603                        MOV     (SP)+,R3        ;RESTORE REGISTERS
 3870  024752  012602                        MOV     (SP)+,R2        ;
 3871  024754  012601                        MOV     (SP)+,R1        ;
 3872  024756  012600                        MOV     (SP)+,R0        ;
 3873  024760  062706  000010                ADD     #8.,SP          ;POP SECOND ARGUMENT
 3874  024764  105737  002400                TSTB    $FPS            ;FOR D MODE?
 3875  024770  100404                        BMI     26$             ;D MODE
 3876  024772  012666  000002                MOV     (SP)+,2(SP)     ;F MODE - CONVERT
 3877  024776  012666  000002                MOV     (SP)+,2(SP)     ;
 3878  025002  000134                  26$:  JMP     @(R4)+          ;DONE
 3879
 3880
 3881  025004  005700                  14$:  TST     R0              ;CHECK HIGH ORDER FRACTION RESULT
 3882  025006  003014                        BGT     22$             ;IF + SIGN OK
 3883  025010  001453                        BEQ     23$             ;CHECK FOR ZERO RESULT
 3884  025012  005403                        NEG     R3              ;ABS VALUE
 3885  025014  005502                        ADC     R2              ;
 3886  025016  005501                        ADC     R1              ;
 3887  025020  005500                        ADC     R0              ;
 3888  025022  005402                        NEG     R2              ;
 3889  025024  005501                        ADC     R1              ;
```

```
3890  025026  005500                        ADC     R0              ;
3891  025030  005401                        NEG     R1              ;
3892  025032  005500                        ADC     R0              ;
3893  025034  005400                        NEG     R0              ;
3894  025036  000316                        SWAB    (SP)            ;EXCHANGE SIGNS
3895  025040  030027  000400          22$:  BIT     R0,#400         ;CHECK NORMAL BIT
3896  025044  001427                        BEQ     19$             ;JUMP IF NONE
3897  025046  005704                        TST     R4              ;CHECK FOR UNDERFLOW
3898  025050  003203                        BGT     9$              ;
3899  025052  032737  002000  002400        BIT     #002000,$FPS    ;UNDERFLOW - IS IT ENABLED ?
3900  025060  001781                        BEQ     33$             ;NO, MAKE ZERO RESULT
3901  025062  005237  002544                INC     ADDC7           ;C1R1  UNDERFLOW
3902  025066  005237  002546                INC     ADDC8           ;C1R1  UNDERFLOW, ENABLED
3903  025072  052737  100000  002400        BIS     #100000,$FPS    ;SET ERROR BIT
3904  025100  012737  000012  002402        MOV     #12,$FEC        ;SET $FEC
3905  025106  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
3906  025114  000304                        SWAB    R4              ;MOVE EXP LEFT
3907  025116  105004                        CLRB    R4              ;CLEAR WHERE FRACTION WILL GO
3908  025120  000137  024536                JMP     16$             ;ASSEMBLE NUMBER
3909  025124  005304                  19$:  DEC     R4              ;DECREASE EXP
3910  025126  006303                        ASL     R3              ;DOUBLE FRACTION
3911  025130  006102                        ROL     R2              ;
3912  025132  006101                        ROL     R1              ;
3913  025134  006100                        ROL     R0              ;
3914  025136  002740                        BP      22$             ;
3915  025140  162704  000010          23$:  SUB     #8.,R4          ;REDUCE EXP
3916  025144  005701                        TST     R1              ;
3917  025146  001020                        BNE     24$             ;JUMP IF ONLY R0=0
3918  025150  162704  000020                SUB     #16.,R4         ;
3919  025154  010201                        MOV     R2,R1           ;
3920  025156  001012                        BNE     25$             ;JUMP IF R2 NON ZERO
3921  025160  162704  000020                SUB     #16.,R4         ;
3922  025164  005703                        TST     R3              ;
3923  025166  001422                        BEQ     21$             ;ANSWER IS ZERO
3924  025170  150301                        BISB    R3,R1           ;MOVE BYTES TO R0,R1
3925  025172  000301                        SWAB    R1              ;
3926  025174  000303                        SWAB    R3              ;
3927  025176  150300                        BISB    R3,R0           ;
3928  025200  005003                        CLR     R3              ;MAKE ALL OTHERS ZERO
3929  025202  000715                        BR      22$             ;GO NORMALIZE
3930  025204  010302                  25$:  MOV     R3,R2           ;
3931  025206  005003                        CLR     R3              ;
3932  025210  000301                  24$:  SWAB    R1              ;MOVE LEFT
3933  025212  150100                        BISB    R1,R0           ;
3934  025214  105001                        CLRB    R1              ;
3935  025216  000302                        SWAB    R2              ;
3936  025220  150201                        BISB    R2,R1           ;
3937  025222  105002                        CLRB    R2              ;
3938  025224  000303                        SWAB    R3              ;
3939  025226  150302                        BISB    R3,R2           ;
3940  025230  105003                        CLRB    R3              ;
3941  025232  000702                        BR      22$             ;
3942  025234  005016                  21$:  CLR     (SP)            ;SET POSITIVE
3943  025236  005004                        CLR     R4              ;
3944  025240  000137  024536                JMP     16$             ;
3945                                        ;*END OF ADD/SUB
```



```
3946                                        ;*FLOATING POINT SOFTWARE MULTIPLY ROUTINE
3947                                        ;*
3948                                        ;*  THIS ROUTINE TAKES THE TOP TWO ELEMENTS ON THE STACK
3949                                        ;*    (LENGTH OF 2 OR 4 WORDS DEPENDING UPON WHETHER THE
3950                                        ;*    F/D MODE BIT7 IN $FPS IS 0 OR 1 RESPECTIVELY) AND
3951                                        ;*    REPLACES THEM WITH THEIR PRODUCT.
3952                                        ;*
3953                                        ;*  EXAMPLE:    $PUSH
3954                                        ;*              ADDR(OPERAND A)
3955                                        ;*              $PUSH
3956                                        ;*              ADDR(OPERAND B)
3957                                        ;*              $MUL
3958                                        ;*              $POPX
3959                                        ;*              ADDR(RESULT)
3960                                        ;*                RESULT=A*B
3961                                        ;*
3962                                        ;*  NOTE ROUTINE IS CALLED THROUGH POLISH MODE OPERATORS.
3963                                        ;*
3964                                        ;*  ALL OPERATIONS ARE CARRIED OUT IN DOUBLE (LENGTH=4)
3965                                        ;*    PRECISION MODE.  SINGLE PRECISION OPERANDS ARE
3966                                        ;*    EXTENDED (VIA $CONV SUBROUTINE) WITH LOW-ORDER
3967                                        ;*    ZEROS, AND LATER TRUNCATED BACK TO SINGLE PRECISION
3968                                        ;*    FROM THE DOUBLE PRECISION RESULT.
3969                                        ;*
3970                                        ;*  STATUS BITS:
3971                                        ;*    THE N, Z, V, AND C BITS OF $FPS ARE SET AS FOLLOWS:
3972                                        ;*    N = 1  IF RESULT NEGATIVE (IE, BIT15 = 1),
3973                                        ;*           ELSE  N = 0
3974                                        ;*    Z = 1  IF RESULT ZERO (IE, EXPONENT = 000(8)),
3975                                        ;*           ELSE  Z = 0
3976                                        ;*    V = 1  IF ARITHMETIC OVERFLOW OCCURRED,
3977                                        ;*           ELSE  V = 0
3978                                        ;*    C = 0  ALWAYS
3979                                        ;*
3980                                        ;*  ERROR CONDITIONS:
3981                                        ;*    IF AN ARITHMETIC OVERFLOW CONDITION OCCURS, THE RESULT
3982                                        ;*    WILL BE SET TO ZERO IF THE OVERFLOW ENABLE BIT OF $FPS
3983                                        ;*    (BIT09) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
3984                                        ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE UNDERBIASED
3985                                        ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
3986                                        ;*    $FEC WILL BE SET TO 10(8), AND $FEA WILL BE LOADED WITH
3987                                        ;*    THE VALUE IN LOCATION "EXPFEA".  IN EITHER INSTANCE,
3988                                        ;*    THE V-BIT (BIT01) WILL BE SET.
3989                                        ;*    IF AN ARITHMETIC UNDERFLOW CONDITION OCCURS, THE RESULT
3990                                        ;*    WILL BE SET TO ZERO IF THE UNDERFLOW ENABLE BIT OF $FPS
3991                                        ;*    (BIT10) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
3992                                        ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE OVERBIASED
3993                                        ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
3994                                        ;*    $FEC WILL BE SET TO 12(8), AND $FEA WILL BE LOADED WITH
3995                                        ;*    THE VALUE IN LOCATION "EXPFEA".
3996                                        ;*
3997                                        ;*  ALL REGISTERS ARE PRESERVED UPON ENTRY, AND RESTORED UPON
3998                                        ;*    EXIT.  THE ROUTINES ARE RE-ENTRANT.
3999                                        ;*
4000                                        ;*  ADAPTED FROM PDP-11 FORTRAN SOFTWARE
4001                                        ;*    BY DONALD NORTH, SEPTEMBER, 1975.
```

```
4002                                                    ;*
4003
4004  025244  005237  002550          $MUL:  INC     MULC0           ;CTR:  TOTAL NUMBER MUL
4005  025250  042737  000017  002400         BIC     #17,$FPS        ;CLEAR STATUS BITS N,Z,V,C
4006  025256  105737  002400                 TSTB    $FPS            ;TEST MODE
4007  025262  100402                         BMI     15$             ;D-MODE
4008  025264  004737  027622                 JSR     PC,$CONV        ;F-MODE:  CONVERT 2 F OPDS TO D
4009  025270  010046                 15$:    MOV     R0,-(SP)        ;
4010  025272  010146                         MOV     R1,-(SP)        ;
4011  025274  010246                         MOV     R2,-(SP)        ;SAVE ALL REGISTERS
4012  025276  010346                         MOV     R3,-(SP)        ;
4013  025300  010446                         MOV     R4,-(SP)        ;
4014  025302  010546                         MOV     R5,-(SP)        ;
4015  025304  006366  000014                 ASL     14(SP)          ;SHIFT MULTIPLICAND
4016  025310  006146                         ROL     -(SP)           ;KEEP SIGN
4017  025312  005046                         CLR     -(SP)           ;CLEAR EXPONENT
4018  025314  116616  000021                 MOVB    21(SP),(SP)     ;KEEP MULTIPLICAND EXP
4019  025320  001436                         BEQ     1$              ;ANSWER ZERO
4020  025322  116666  000020  000021         MOVB    20(SP),21(SP)   ;SHIFT FRACTION LEFT
4021  025330  000261                         SEC                     ;INSERT NORMAL BIT.
4022  025332  006066  000020                 ROR     20(SP)          ;
4023  025336  116666  000023  000020         MOVB    23(SP),20(SP)   ;
4024  025344  006366  000022                 SWAB    22(SP)          ;
4025  025350  116666  000025  000022         MOVB    25(SP),22(SP)   ;
4026  025356  006366  000024                 SWAB    24(SP)          ;
4027  025362  116666  000027  000024         MOVB    27(SP),24(SP)   ;
4028  025370  006366  000026                 SWAB    26(SP)          ;
4029  025374  105066  000026                 CLRB    26(SP)          ;EXTRA BITS
4030  025400  006366  000030                 ASL     30(SP)          ;SHIFT HIGH MULTIPLIER
4031  025404  005566  000002                 ADC     2(SP)           ;PRODUCT SIGN
4032  025410  105766  000031                 TSTB    31(SP)          ;
4033  025414  001005                         BNE     2$              ;JUMP IF NON ZERO
4034  025416  022626                 1$:     CMP     (SP)+,(SP)+     ;FLUSH SIGN AND EXPONENT
4035  025420  005237  002552                 INC     MULC1           ;CTR:  A=0 AND/OR B=0
4036  025424  000137  026100                 JMP     3$              ;
4037  025430  005000                 2$:     CLR     R0              ;CLEAR PRODUCT
4038  025432  005001                         CLR     R1              ;
4039  025434  005002                         CLR     R2              ;
4040  025436  005003                         CLR     R3              ;
4041  025440  005005                         CLR     R5              ;CLEAR C BIT OVERFLOW CATCHER
4042  025442  006066  000030                 ROR     30(SP)          ;SIGN IS +
4043  025446  012746  000020                 MOV     #16.,-(SP)      ;ITERATION COUNT
4044  025452  016604  000040                 MOV     40(SP),R4       ;GET LOWEST ORDER MULTIPLIER
4045  025456  001404                         BEQ     4$              ;JUMP IF NO BITS HERE
4046  025460  004737  026320                 JSR     PC,30$          ;
4047  025464  012716  000020                 MOV     #16.,(SP)       ;RESTORE COUNT
4048  025470  016604  000036         4$:     MOV     36(SP),R4       ;GET NEXT LOWEST FRACTION
4049  025474  001003                         BNE     5$              ;WORK TO DO
4050  025476  005766  000040                 TST     40(SP)          ;
4051  025502  001406                         BEQ     6$              ;NO PRODUCT YET
4052  025504  004737  026314         5$:     JSR     PC,31$          ;
4053  025510  004737  026222                 JSR     PC,32$          ;ONE BIT FULL PRECISION
4054  025514  012716  000020                 MOV     #16.,(SP)       ;
4055  025520  016604  000034         6$:     MOV     34(SP),R4       ;NEXT TO HIGHEST ORDER FRACTION
4056  025524  001006                         BNE     7$              ;
4057  025526  005766  000036                 TST     36(SP)          ;
```



```
4058  025532  001003                         BNE     7$              ;
4059  025534  005766  000040                 TST     40(SP)          ;
4060  025540  001402                         BEQ     8$              ;
4061  025542  004737  026222         7$:     JSR     PC,32$          ;
4062  025546  016604  000032         8$:     MOV     32(SP),R4       ;GET HIGH ORDER BITS
4063  025552  012716  000007                 MOV     #7,(SP)         ;SEVEN OF THEM
4064  025556  004737  026222                 JSR     PC,32$          ;
4065  025562  004737  026226                 JSR     PC,33$          ;NORMAL BIT
4066  025566  005726                         TST     (SP)+           ;FLUSH ITERATION COUNT
4067  025570  062604                         ADD     (SP)+,R4        ;ADD EXP
4068  025572  006303                         ASL     R3              ;SHIFT OUT NORMAL BIT
4069  025574  006102                         ROL     R2              ;
4070  025576  006101                         ROL     R1              ;
4071  025600  006100                         ROL     R0              ;
4072  025602  103405                         BCS     9$              ;NORMAL BIT FOUND
4073  025604  006303                         ASL     R3              ;
4074  025606  006102                         ROL     R2              ;
4075  025610  006101                         ROL     R1              ;
4076  025612  006100                         ROL     R0              ;HAVE IT
4077  025614  005304                         DEC     R4              ;ADJUST EXP
4078  025616  162704  000200         9$:     SUB     #200,R4         ;REMOVE BIAS FROM EXP
4079  025622  003022                         BGT     10$             ;BR IF NO UNDERFLOW
4080  025624  005237  002560                 INC     MULC4           ;CTR:  UNDERFLOW
4081  025630  032737  002000  002400         BIT     #2000,$FPS      ;UNDERFLOW - IS IT ENABLED ?
4082  025636  001517                         BEQ     12$             ;NO, MAKE ZERO RESULT
4083  025640  005237  002562                 INC     MULC5           ;CTR:  UNDERFLOW, ENABLED
4084  025644  052737  100000  002400         BIS     #100000,$FPS    ;SET ERROR BIT
4085  025652  012737  000012  002402         MOV     #12,$FEC        ;SET $FEC
4086  025660  013737  002376  002404         MOV     EXPFEA,$FEA     ;SET $FEA
4087  025666  000427                         BR      11$             ;CONTINUE
4088  025670  022704  000377         10$:    CMP     #377,R4         ;CHECK FOR OVERFLOW
4089  025674  002024                         BGE     11$             ;BR IF NO OVERFLOW
4090  025676  052737  000002  002400         BIS     #2,$FPS         ;SET V BIT ON OVERFLOW
4091  025704  005237  002554                 INC     MULC2           ;CTR:  OVERFLOW
4092  025710  032737  001000  002400         BIT     #1000,$FPS      ;OVERFLOW ENABLED ?
4093  025716  001467                         BEQ     12$             ;NO, ZERO RESULT
4094  025720  005237  002556                 INC     MULC3           ;CTR:  OVERFLOW, ENABLED
4095  025724  052737  100000  002400         BIS     #100000,$FPS    ;SET ERROR BIT
4096  025732  012737  000010  002402         MOV     #10,$FEC        ;SET $FEC
4097  025740  013737  002376  002404         MOV     EXPFEA,$FEA     ;SET $FEA
4098  025746  105003                 11$:    CLRB    R3              ;
4099  025750  152203                         BISB    R2,R3           ;SHIFT FRACTION RIGHT
4100  025752  006303                         SWAB    R3              ;
4101  025754  105002                         CLRB    R2              ;
4102  025756  152102                         BISB    R1,R2           ;
4103  025760  006302                         SWAB    R2              ;
4104  025762  105001                         CLRB    R1              ;
4105  025764  152001                         BISB    R0,R1           ;
4106  025766  006301                         SWAB    R1              ;
4107  025770  105000                         CLRB    R0              ;
4108  025772  150400                         BISB    R4,R0           ;
4109  025774  006300                         SWAB    R0              ;
4110  025776  006026                         ROR     (SP)+           ;GET PRODUCT SIGN
4111  026000  006000                         ROR     R0              ;PUT IN RESULT
4112  026002  006001                         ROR     R1              ;
4113  026004  006002                         ROR     R2              ;
```

```
4114  026006  006003                      ROR     R3              ;
4115  026010  005503                      ADC     R3              ;ROUND RESULT
4116  026012  005502                      ADC     R2              ;
4117  026014  005501                      ADC     R1              ;
4118  026016  005500                      ADC     R0              ;
4119  026020  103401                      BCS     16$             ;OVERFLOW ON ROUND ?
4120  026022  102032                      BVC     13$             ;
4121  026024  052737  000002  002404  16$:  BIS     #02,$FPS        ;YES - SET V BIT
4122  026032  005237  002554              INC     MULC2           ;CTR:  OVERFLOW
4123  026036  032737  001000  002404      BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
4124  026044  001415                      BEQ     3$              ;NO, ZERO RESULT
4125  026046  005237  002556              INC     MULC3           ;CTR:  OVERFLOW, ENABLED
4126  026052  052737  100000  002404      BIS     #100000,$FPS    ;SET ERROR BIT
4127  026060  012737  000010  002402      MOV     #10,$FEC        ;SET $FEC
4128  026066  013737  002376  002404      MOV     FXPFEA,$FEA     ;SET $FEA
4129  026074  000405                      BR      13$             ;CONTINUE
4130  026076  005726              12$:    TST     (SP)+           ;FLUSH SIGN
4131  026100  005000              3$:     CLR     R0              ;CLEAR RESULT
4132  026102  005001                      CLR     R1              ;
4133  026104  005002                      CLR     R2              ;
4134  026106  005003                      CLR     R3              ;
4135  026110  010066  000024      13$:    MOV     R0,24(SP)       ;
4136  026114  010166  000026              MOV     R1,26(SP)       ;STUFF RESULT
4137  026120  010266  000030              MOV     R2,30(SP)       ;
4138  026124  010366  000032              MOV     R3,32(SP)       ;
4139  026130  032766  077600  000024      BIT     #077600,24(SP)  ;SET Z BIT IF EXPONENT ZERO
4140  026136  001003                      BNE     17$             ;
4141  026140  052737  000004  002400      BIS     #04,$FPS        ;
4142  026146  005766  000024              TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
4143  026152  100003                      BPL     18$             ;
4144  026154  052737  000010  002400      BIS     #10,$FPS        ;
4145  026162  012605              19$:    MOV     (SP)+,R5        ;
4146  026164  012604                      MOV     (SP)+,R4        ;
4147  026166  012603                      MOV     (SP)+,R3        ;RESTORE REGISTERS
4148  026170  012602                      MOV     (SP)+,R2        ;
4149  026172  012601                      MOV     (SP)+,R1        ;
4150  026174  012600                      MOV     (SP)+,R0        ;
4151  026176  062706  000010              ADD     #8.,SP          ;CLEAR SECOND OPERAND OFF STACK
4152  026202  105737  002400              TSTB    $FPS            ;F OR D MODE
4153  026206  100404                      BMI     14$             ;D-MODE
4154  026210  012666  000002              MOV     (SP)+,2(SP)     ;F MODE - CONVERT
4155  026214  012666  000002              MOV     (SP)+,2(SP)     ;
4156  026220  000134              14$:    JMP     @(R4)+          ;RETURN
4157
4158  026222  006204              32$:    ASR     R4              ;TEST NEXT MULTIPLIER BIT
4159  026224  103422                      BCC     34$             ;JUMP IF ZERO
4160  026226  066603  000032      33$:    ADD     32(SP),R3       ;ADD IN MULTIPLICAND
4161  026232  005502                      ADC     R2              ;
4162  026234  005501                      ADC     R1              ;
4163  026236  005500                      ADC     R0              ;
4164  026240  005505                      ADC     R5              ;SAVE OVERFLOW
4165  026242  066602  000030              ADD     30(SP),R2       ;
4166  026246  005501                      ADC     R1              ;
4167  026250  005500                      ADC     R0              ;
4168  026252  005505                      ADC     R5              ;
4169  026254  066601  000026              ADD     26(SP),R1       ;
```



```
4170  026260  005503                      ADC     R0              ;
4171  026262  005505                      ADC     R5              ;
4172  026264  066600  000024              ADD     24(SP),R0       ;
4173  026270  005505                      ADC     R5              ;
4174  026272  006205              34$:    ASR     R5              ;RECOVER OVERFLOW IF ANY
4175  026274  006000                      ROR     R0              ;SHIFT PRODUCT
4176  026276  006001                      ROR     R1              ;
4177  026300  006002                      ROR     R2              ;
4178  026302  006003                      ROR     R3              ;
4179  026304  005366  000002              DEC     2(SP)           ;COUNT LOOP
4180  026310  003344                      BGT     32$             ;AGAIN
4181  026312  000207                      RTS     PC              ;RETURN
4182  026314  005366  177002      31$:    DEC     2(SP)           ;ONLY 15 BITS THIS PASS
4183  026320  006204              30$:    ASR     R4              ;TEST NEXT MULTIPLIER BIT
4184  026322  103007                      BCC     35$             ;JUMP IF ZERO
4185  026324  066601  000026              ADD     26(SP),R1       ;USE ONLY HIGH ORDER MULTIPLICAND
4186  026330  005500                      ADC     R0              ;
4187  026332  005505                      ADC     R5              ;
4188  026334  066600  000024              ADD     24(SP),R0       ;
4189  026340  005505                      ADC     R5              ;
4190  026342  006205              35$:    ASR     R5              ;RECOVER ANY OVERFLOW
4191  026344  006000                      ROR     R0              ;
4192  026346  006001                      ROR     R1              ;
4193  026350  006002                      ROR     R2              ;
4194  026352  006003                      ROR     R3              ;
4195  026354  005366  000002              DEC     2(SP)           ;COUNT LOOP
4196  026360  003357                      BGT     30$             ;
4197  026362  000207                      RTS     PC              ;RETURN
4198                                      ;*END OF MUL
```

```
                                        ;*FLOATING POINT SOFTWARE DIVIDE ROUTINE
                                        ;*
                                        ;*  THIS ROUTINE TAKES THE TOP TWO ELEMENTS ON THE STACK
                                        ;*    (LENGTH OF 2 OR 4 WORDS DEPENDING UPON WHETHER THE
                                        ;*    F/D MODE BIT7 IN $FPS IS 0 OR 1 RESPECTIVELY) AND
                                        ;*    REPLACES THEM WITH THEIR QUOTIENT.
                                        ;*
                                        ;*  EXAMPLE:    $PUSH
                                        ;*              ADDR(OPERAND A)
                                        ;*              $PUSH
                                        ;*              ADDR(OPERAND B)
                                        ;*              $DIV
                                        ;*              $POPX
                                        ;*              ADDR(RESULT)
                                        ;*                RESULT=A/B
                                        ;*
                                        ;*  NOTE ROUTINE IS CALLED THROUGH POLISH MODE OPERATORS.
                                        ;*
                                        ;*  ALL OPERATIONS ARE CARRIED OUT IN DOUBLE (LENGTH=4)
                                        ;*    PRECISION MODE.  SINGLE PRECISION OPERANDS ARE
                                        ;*    EXTENDED (VIA $CONV SUBROUTINE) WITH LOW-ORDER
                                        ;*    ZEROS, AND LATER TRUNCATED BACK TO SINGLE PRECISION
                                        ;*    FROM THE DOUBLE PRECISION RESULT.
                                        ;*
                                        ;*  STATUS BITS:
                                        ;*    THE N, Z, V, AND C BITS OF $FPS ARE SET AS FOLLOWS:
                                        ;*    N = 1  IF RESULT NEGATIVE (IE, BIT15 = 1),
                                        ;*           ELSE  N = 0
                                        ;*    Z = 1  IF RESULT ZERO (IE, EXPONENT = 000(8)),
                                        ;*           ELSE  Z = 0
                                        ;*    V = 1  IF ARITHMETIC OVERFLOW OCCURRED,
                                        ;*           ELSE  V = 0
                                        ;*    C = 0  ALWAYS
                                        ;*
                                        ;*  ERROR CONDITIONS:
                                        ;*    IF AN ARITHMETIC OVERFLOW CONDITION OCCURS, THE RESULT
                                        ;*    WILL BE SET TO ZERO IF THE OVERFLOW ENABLE BIT OF $FPS
                                        ;*    (BIT09) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
                                        ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE UNDERBIASED
                                        ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
                                        ;*    $FEC WILL BE SET TO 10(8), AND $FEA WILL BE LOADED WITH
                                        ;*    THE VALUE IN LOCATION "EXPFEA".  IN EITHER INSTANCE,
                                        ;*    THE V-BIT (BIT01) WILL BE SET.
                                        ;*    IF AN ARITHMETIC UNDERFLOW CONDITION OCCURS, THE RESULT
                                        ;*    WILL BE SET TO ZERO IF THE UNDERFLOW ENABLE BIT OF $FPS
                                        ;*    (BIT10) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
                                        ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE OVERBIASED
                                        ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
                                        ;*    $FEC WILL BE SET TO 12(8), AND $FEA WILL BE LOADED WITH
                                        ;*    THE VALUE IN LOCATION "EXPFEA".
                                        ;*    IF DIVISION BY ZERO IS ATTEMPTED (EG, EXPONENT OF
                                        ;*    DENOMINATOR OPERAND IS ZERO), THE RESULT LEFT ON THE
                                        ;*    STACK WILL BE THE NUMERATOR, WITH THE CONDITION CODES SET
                                        ;*    ACCORDINGLY.  THE $FPS ERROR BIT WILL BE SET, $FEC WILL BE
                                        ;*    SET TO 4(8), AND $FEA WILL BE SET TO THE VALUE CONTAINED
                                        ;*    IN THE LOCATION "EXPFEA".
```



```
                                        ;*
                                        ;*  ALL REGISTERS ARE PRESERVED UPON ENTRY, AND RESTORED UPON
                                        ;*    EXIT.  THE ROUTINES ARE RE-ENTRANT.
                                        ;*
                                        ;*  ADAPTED FROM PDP-11 FORTRAN SOFTWARE
                                        ;*    BY DONALD NORTH, SEPTEMBER, 1975.
                                        ;*
001026 026364 005237 002564     $DIV:   INC     DIVC0           ;C1R:  TOTAL NUMBER OF DIV
001032 026370 042737 000017 002400      BIC     #17,$FPS        ;CLEAR STATUS BITS N Z V C
001040 026376 105737 002400             TSTB    $FPS            ;TEST MODE
001044 026402 100402                    BMI     14$             ;D-MODE
001046 026404 004737 027622             JSR     PC,$CONV        ;F-MODE:  CONVERT 2 F OPDS TO D
001052 026410 010046            14$:    MOV     R0,-(SP)        ;
001054 026412 010146                    MOV     R1,-(SP)        ;
001056 026414 010246                    MOV     R2,-(SP)        ;
001060 026416 010346                    MOV     R3,-(SP)        ;SAVE REGISTERS
001062 026420 010446                    MOV     R4,-(SP)        ;
001064 026422 010546                    MOV     R5,-(SP)        ;
001066 026424 032766 077600 000014      BIT     #0776R0,14(SP)  ;DIVIDE BY ZERO ?
001074 026432 001015                    BNE     2$              ;NO
001076 026434 005237 002570             INC     DIVC2           ;CTR:  DENOM=0
001102 026440 052737 100000 002400      BIS     #100000,$FPS    ;YES, SET ERROR BIT
001110 026446 012737 000004 002402      MOV     #4,$FEC         ;SET $FEC
001116 026454 013737 002376 002404      MOV     EXPFEA,$FEA     ;SET $FEA
001124 026462 000137 027372             JMP     9$              ;DONE
001130 026466 005000            2$:     CLR     R0              ;
001132 026470 005001                    CLR     R1              ;
001134 026472 005002                    CLR     R2              ;
001136 026474 005003                    CLR     R3              ;
001140 026476 005046                    CLR     -(SP)           ;
001142 026500 006366 000026             ASL     26(SP)          ;SHIFT NUMERATOR
001146 026504 006116                    ROL     (SP)            ;NUMERATOR SIGN
001150 026506 005046                    CLR     -(SP)           ;
001152 026510 156616 000031             BISB    31(SP),(SP)     ;NUMERATOR EXP
001156 026514 001004                    BNE     6$              ;NUMERATOR IS ZERO?
001160 026516 005237 002566             INC     DIVC1           ;CTR:  NUMER=0
001164 026522 000137 027346             JMP     1$              ;YES, DONE
001170 026526 156600 000030     6$:     BISB    30(SP),R0       ;
001174 026532 000300                    SWAB    R0              ;LEFT JUSTIFY NUMERATOR FRACTION
001176 026534 000261                    SEC                     ;INSERT NORMAL BIT
001200 026536 006000                    ROR     R0              ;
001202 026540 156600 000033             BISB    33(SP),R0       ;
001206 026544 156601 000032             BISB    32(SP),R1       ;
001212 026550 000301                    SWAB    R1              ;
001214 026552 156601 000035             BISB    35(SP),R1       ;
001220 026556 156602 000034             BISB    34(SP),R2       ;
001224 026562 000302                    SWAB    R2              ;
001226 026564 156602 000037             BISB    37(SP),R2       ;
001232 026570 156603 000036             BISB    36(SP),R3       ;
001236 026574 000303                    SWAB    R3              ;
001240 026576 006366 000020             ASL     20(SP)          ;SHIFT DENOMINATOR
001244 026602 005566 000002             ADC     2(SP)           ;RESULT SIGN
001250 026606 005004                    CLR     R4              ;
001252 026610 156604 000021             BISB    21(SP),R4       ;DIVISOR EXPONENT
001256 026614 160416                    SUB     R4,(SP)         ;SUBTRACT EXPONENTS
```

```
 4311  026610  000366  000020            SWAB    20(SP)          ;LEFT JUSTIFY DENOM
 4312  026622  000261                    SEC                     ;INSERT NORMAL BIT
 4313  026624  006066  000020            ROR     20(SP)          ;
 4314  026630  116666  000023  000020    MOVB    23(SP),20(SP)   ;
 4315  026636  116666  000022  000023    MOVB    22(SP),23(SP)   ;
 4316  026644  116666  000025  000022    MOVB    25(SP),22(SP)   ;
 4317  026652  116666  000024  000025    MOVB    24(SP),25(SP)   ;
 4318  026660  116666  000027  000024    MOVB    27(SP),24(SP)   ;
 4319  026666  116666  000026  000027    MOVB    26(SP),27(SP)   ;
 4320  026674  105066  000026            CLRB    26(SP)          ;
 4321  026700  005066  000030            CLR     30(SP)          ;CLEAR QUOTIENT
 4322  026704  005066  000032            CLR     32(SP)          ;
 4323  026710  005066  000034            CLR     34(SP)          ;
 4324  026714  020066  000020            CMP     R0,20(SP)       ;COMPARE HIGH NUM + DENOM
 4325  026720  101020                    BHI     3$              ;JUMP IF DENOM LOW
 4326  026722  103424                    BLO     4$              ;JUMP IF DENOM HI
 4327  026724  020166  000022            CMP     R1,22(SP)       ;COMPARE LOW ORDER PARTS
 4328  026730  101014                    BHI     3$              ;
 4329  026732  103420                    BLO     4$              ;
 4330  026734  020266  000024            CMP     R2,24(SP)       ;
 4331  026740  101010                    BHI     3$              ;
 4332  026742  103414                    BLO     4$              ;
 4333  026744  020366  000026            CMP     R3,26(SP)       ;
 4334  026750  101004                    BHI     3$              ;
 4335  026752  001010                    BNE     4$              ;
 4336  026754  005216                    INC     (SP)            ;BUMP EXP
 4337  026756  005004                    CLR     R4              ;
 4338  026760  000443                    BR      5$              ;
 4339
 4340  026762  006000            3$:     ROR     R0              ;HALVE DENOM (C=0)
 4341  026764  006001                    ROR     R1              ;TO INSURE N<D
 4342  026766  006002                    ROR     R2              ;
 4343  026770  006003                    ROR     R3              ;
 4344  026772  005216                    INC     (SP)            ;COMPENSATE EXP
 4345  026774  012705  000011    4$:     MOV     #9.,R5          ;FIRST NINE QUOTIENT BITS
 4346  027000  004737  027464            JSR     PC,30$          ;
 4347  027004  110466  000030            MOVB    R4,30(SP)       ;SAVE ALL HIGH ORDER Q FRACTION
 4348  027010  005705                    TST     R5              ;DONE?
 4349  027012  001025                    BNE     10$             ;YES - REST OF NUMBER IS 0
 4350  027014  012705  000020            MOV     #16.,R5         ;16 MORE BITS
 4351  027020  004737  027464            JSR     PC,30$          ;
 4352  027024  110466  000032            MOV     R4,32(SP)       ;
 4353  027030  005705                    TST     R5              ;
 4354  027032  001015                    BNE     10$             ;
 4355  027034  012705  000020            MOV     #16.,R5         ;
 4356  027040  004737  027464            JSR     PC,30$          ;
 4357  027044  010466  000034            MOV     R4,34(SP)       ;
 4358  027050  005705                    TST     R5              ;
 4359  027052  001005                    BNE     10$             ;
 4360  027054  012705  000020            MOV     #16.,R5         ;
 4361  027060  004737  027464            JSR     PC,30$          ;
 4362  027064  000401                    BR      5$              ;
 4363  027066  005004            10$:    CLR     R4              ;CLEAR LOWEST ORDER QUOTIENT
 4364  027070  012605            5$:     MOV     (SP)+,R5        ;PUSH UP EXPONENT
 4365  027072  062705  000200            ADD     #200,R5         ;INSERT BIAS
 4366  027076  003022                    BGT     7$              ;BR IF NO UNDERFLOW
```



```
 4367  027100  005237  002576            INC     DIVC5           ;CTR:  UNDERFLOW
 4368  027104  032737  002000  002400    BIT     #002000,$FPS    ;UNDERFLOW - IS IT ENABLED ?
 4369  027112  001516                    BEQ     15$             ;NO, MAKE ZERO RESULT
 4370  027114  005237  002600            INC     DIVC6           ;CTR:  UNDERFLOW, ENABLED
 4371  027120  052737  100000  002400    BIS     #100000,$FPS    ;SET ERROR BIT
 4372  027126  012737  000012  002402    MOV     #12,$FEC        ;SET $FEC
 4373  027134  013737  002376  002404    MOV     EXPFEA,$FEA     ;SET $FEA
 4374  027142  000427                    BR      11$             ;CONTINUE
 4375  027144  022705  000377    7$:     CMP     #377,R5         ;CHECK FOR OVERFLOW
 4376  027150  002024                    BGE     11$             ;BR IF NO OVERFLOW
 4377  027152  052737  000002  002400    BIS     #02,$FPS        ;SET V BIT ON OVERFLOW
 4378  027160  005237  002572            INC     DIVC3           ;CTR:  OVERFLOW
 4379  027164  032737  001000  002400    BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
 4380  027172  001466                    BEQ     15$             ;NO, ZERO RESULT
 4381  027174  005237  002574            INC     DIVC4           ;CTR:  OVERFLOW, ENABLED
 4382  027200  052737  100000  002400    BIS     #100000,$FPS    ;SET ERROR BIT
 4383  027206  012737  000010  002402    MOV     #10,$FEC        ;SET $FEC
 4384  027214  013737  002376  002404    MOV     EXPFEA,$FEA     ;SET $FEA
 4385  027222  110566  000027    11$:    MOVB    R5,27(SP)       ;PUT EXPIN RESULT
 4386  027226  006026                    ROR     (SP)+           ;INSERT SIGN
 4387  027230  006066  000024            ROR     24(SP)          ;
 4388  027234  006066  000026            ROR     26(SP)          ;
 4389  027240  006066  000030            ROR     30(SP)          ;
 4390  027244  006004                    ROR     R4              ;
 4391  027246  005504                    ADC     R4              ;ROUND
 4392  027250  005566  000030            ADC     30(SP)          ;
 4393  027254  005566  000026            ADC     26(SP)          ;
 4394  027260  005566  000024            ADC     24(SP)          ;
 4395  027264  010466  000032            MOV     R4,32(SP)       ;INSERT LOW ORDER FRACTION
 4396  027270  103401                    BCS     16$             ;OVERFLOW ON ROUND ?
 4397  027272  102037                    BVC     9$              ;
 4398  027274  052737  000002  002400  15$:  BIS   #02,$FPS      ;YES - SET V BIT
 4399  027302  005237  002572            INC     DIVC3           ;CTR:  OVERFLOW
 4400  027306  032737  001000  002400    BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
 4401  027314  001416                    BEQ     19$             ;NO, ZERO RESULT
 4402  027316  005237  002574            INC     DIVC4           ;CTR:  OVERFLOW, ENABLED
 4403  027322  052737  100000  002400    BIS     #100000,$FPS    ;SET ERROR BIT
 4404  027330  012737  000010  002402    MOV     #10,$FEC        ;SET $FEC
 4405  027336  013737  002376  002404    MOV     EXPFEA,$FEA     ;SET $FEA
 4406  027344  000412                    BR      9$              ;CONTINUE
 4407  027346  005726            18:     TST     (SP)+           ;FLUSH EXP
 4408  027350  005726            15$:    TST     (SP)+           ;FLUSH SIGN
 4409  027352  005066  000024    19$:    CLR     24(SP)          ;CLEAR RESULT
 4410  027356  005066  000026            CLR     26(SP)          ;
 4411  027362  005066  000030            CLR     30(SP)          ;
 4412  027366  005066  000032            CLR     32(SP)          ;
 4413  027372  032766  077600  000024  9$:  BIT   #077600,24(SP) ;SET Z BIT IF EXPONENT ZERO
 4414  027400  001003                    BNE     17$             ;
 4415  027402  052737  000004  002400    BIS     #04,$FPS        ;
 4416  027410  005766  000024    17$:    TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
 4417  027414  100003                    BPL     18$             ;
 4418  027416  052737  000010  002400    BIS     #10,$FPS        ;
 4419  027424  012605            16$:    MOV     (SP)+,R5        ;
 4420  027426  012604                    MOV     (SP)+,R4        ;
 4421  027430  012603                    MOV     (SP)+,R3        ;RESTORE REGISTERS
 4422  027432  012602                    MOV     (SP)+,R2        ;
```

```
 4423  027434  012701                      MOV     (SP)+,R1        ;
 4424  027436  012600                      MOV     (SP)+,R0        ;
 4425  027440  062706  000010              ADD     #8.,SP          ;FLUSH FIRST ARG
 4426  027444  105737  002480              TSTB    @FPS            ;F OR D MODE
 4427  027450  100404                      BMI     13$             ;D MODE
 4428  027452  012666  000002              MOV     (SP)+,2(SP)     ;F MODE - CONVERT
 4429  027456  012666  000002              MOV     (SP)+,2(SP)     ;
 4430  027462  000134              13$:    JMP     @(R4)+          ;RETURN
 4431
 4432  027464  006304              30$:    ASL     R4              ;SHIFT QUOTIENT
 4433  027466  006303                      ASL     R3              ;
 4434  027470  006102                      ROL     R2              ;
 4435  027472  006101                      ROL     R1              ;
 4436  027474  006100                      ROL     R0              ;
 4437  027476  103424                      BCS     31$             ;GUARANTEED TO GO
 4438  027500  026600  000022              CMP     22(SP),R0       ;COMPARE HIGH DIVISOR AND DIVIDEND
 4439  027504  101034                      BHI     32$             ;DIVISOR BIGGER
 4440  027506  103414                      BLO     31$             ;
 4441  027510  026601  000024              CMP     24(SP),R1       ;CHECK LOW ORDERS
 4442  027514  101030                      BHI     32$             ;
 4443  027516  103410                      BLO     31$             ;
 4444  027520  026602  000026              CMP     26(SP),R2       ;
 4445  027524  101024                      BHI     32$             ;
 4446  027526  103404                      BLO     31$             ;
 4447  027530  026603  000030              CMP     30(SP),R3       ;
 4448  027534  101020                      BHI     32$             ;
 4449  027536  001422                      BEQ     33$             ;NUMER=DENOM
 4450  027540  166603  000030      31$:    SUB     30(SP),R3       ;N=N-D
 4451  027544  005602                      SBC     R2              ;
 4452  027546  005601                      SBC     R1              ;
 4453  027550  005600                      SBC     R0              ;
 4454  027552  166602  000026              SUB     26(SP),R2       ;
 4455  027556  005601                      SBC     R1              ;
 4456  027560  005600                      SBC     R0              ;
 4457  027562  166601  000024              SUB     24(SP),R1       ;
 4458  027566  005600                      SBC     R0              ;
 4459  027570  166600  000022              SUB     22(SP),R0       ;
 4460  027574  005204                      INC     R4              ;INSERT QUOTIENT BIT
 4461  027576  005305              32$:    DEC     R5              ;COUNT LOOP
 4462  027600  003331                      BGT     30$             ;
 4463  027602  000207                      RTS     PC              ;RETURN
 4464  027604  005204              33$:    INC     R4              ;INSERT LAST 1 BIT IN QUOTIENT
 4465  027606  000401                      BR      34$             ;
 4466  027610  006304              35$:    ASL     R4              ;FINISH OUT QUOTIENT WITH ZEROS
 4467  027612  005305              34$:    DEC     R5              ;
 4468  027614  003375                      BGT     35$             ;
 4469  027616  005205                      INC     R5              ;FLAG NO MORE NUMER
 4470  027620  000207                      RTS     PC              ;RETURN
 4471                                      ;*END OF DIV
```



```
 4472                                      ;*CONVERT TOP 2 OPERANDS ON STACK FROM F MODE
 4473                                      ;*    TO D MODE
 4474                                      ;*
 4475                                      ;*  THIS ROUTINE TAKES THE TOP TWO 2-WORD (SINGLE
 4476                                      ;*    PRECISION) FLOATING POINT NUMBERS ON THE
 4477                                      ;*    STACK AND CONVERTS THEM BOTH TO 4-WORD
 4478                                      ;*    (DOUBLE PRECISION) FORMAT BY APPENDING
 4479                                      ;*    TWO WORDS OF ZEROS AS THE LOW ORDER BIT
 4480                                      ;*    EXTENSION.
 4481                                      ;*
 4482
 4483  027622  005046              $CONV:  CLR     -(SP)           ;CLEAR WORD 3 OF A
 4484  027624  016646  000006              MOV     6(SP),-(SP)     ;MOVE WORD 2 OF A
 4485  027630  016646  000006              MOV     6(SP),-(SP)     ;MOVE WORD 1 OF A
 4486  027634  016646  000006              MOV     6(SP),-(SP)     ;MOVE RETURN ADDR TO TOP
 4487  027640  016666  000020  000014      MOV     20(SP),14(SP)   ;MOVE WORD 2 OF B
 4488  027646  016666  000016  000012      MOV     16(SP),12(SP)   ;MOVE WORD 1 OF B
 4489  027654  005066  000020              CLR     20(SP)          ;CLEAR WORD 4 OF B
 4490  027660  005066  000016              CLR     16(SP)          ;CLEAR WORD 3 OF B
 4491  027664  005066  000010              CLR     10(SP)          ;CLEAR WORD 4 OF A
 4492  027670  000207                      RTS     PC              ;
```

```
                                        SCOPE HANDLER ROUTINE

                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;**************************************************************
                                        ;*THIS POUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($ISTNM) INTO THE DISPLAY REG.(DISPLAY<15:00>)
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*SW08=1         LOOP ON TEST IN "$LPTST"
                                        ;*CALL
                                        ;*       SCOPE           ;;SCOPE=IOT

                                        $SCOPE:
                                        64$:
                                        16$:    BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
                                                BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
                                        $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
                                                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
                                                MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
                                                TST     @#177060        ;;TIME OUT ON XOR?
                                                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
                                                BR      $SVLAD          ;;GO TO THE NEXT TEST
                                        5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
                                                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
                                                BR      7$              ;;LOOP ON THE PRESENT TEST
                                        64$:;*****END OF CODE FOR THE XOR TESTER*****
                                                BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
                                                BEQ     2$              ;;BR IF NO
                                                CMP     $LPTST,$TSTNM   ;;ON THE RIGHT TEST?
                                                BEQ     $OVER           ;;BR IF YES
                                        2$:     TST     $ERFLG          ;;HAS AN ERROR OCCURRED?
                                                BEQ     3$              ;;BR IF NO
                                                CMP     $ERMAX,$EPFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
                                                BHI     3$              ;;BR IF NO
                                                BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
                                                BEQ     4$              ;;BR IF NO
                                        7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
                                                BR      6$OVER
                                        4$:     CLR     $ERFLG          ;;ZERO THE ERROR FLAG
                                                CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
                                                BR      1$              ;;ESCAPE TO THE NEXT TEST
                                        3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
                                                BNE     1$              ;;BR IF YES
                                                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
                                                BEQ     1$              ;;       INHIBIT ITERATIONS
                                                INC     $ICNT           ;;INCREMENT ITERATION COUNT
                                                CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
                                                BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
                                        1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
                                                MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
                                        $SVLAD: INC     $TSTNM          ;;COUNT TEST NUMBERS
                                                MOV     $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
                                                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```



```
                                        SCOPE HANDLER ROUTINE

                                                MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
                                                CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
                                                MOV     #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
                                        $OVER:  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
                                                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
                                                RTI                     ;;FIXES PS
                                        $MXCNT: 400                     ;;MAX. NUMBER OF ITERATIONS
```

```
                                        .SBTTL  ERROR HANDLER ROUTINE
                                
                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO $TYPERR ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1         BELL ON ERROR
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
                                
030150                          $ERROR:
030150  012037  002602                  MOV     R0,ERRG0        ; DISPLAY R0
030154  012137  002604                  MOV     R1,ERRG1        ;         R1
030160  013237  002606                  MOV     R2,ERRG2        ;         R2
030164  010337  002610                  MOV     R3,ERRG3        ;         R3
030170  010437  002612                  MOV     R4,ERRG4        ;         R4
030174  010537  002614                  MOV     R5,ERRG5        ;         R5
030200  010637  002616                  MOV     R6,ERRG6        ; GET R6(SP) BEFORE TRAP
030204  062737  000004  002616          ADD     #4,ERRG6        ;
030212  011637  002620                  MOV     (SP),ERRG7      ; PC => ERROR CALL INSTR
030216  005237  001104          7$:     INC     $ERFLG          ;;SET THE ERROR FLAG
030222  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
030224  013777  001102  150714          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
030232  032777  002000  150704          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
030240  001402                  BEQ     1$              ;;NO - SKIP
030242  104401  001172                  TYPE    ,$BELL          ;;RING BELL
030246  005237  001114          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
030252  011637  001122                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
030256  162737  000002  001122          SUB     #2,$ERRPC
030264  117737  150632  001116          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
030272  032777  020000  150644          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
030300  001004                  BNE     2$              ;;SKIP TYPEOUTS
030302  004737  030412                  JSR     PC,$TYPERR      ;;GO TO USER ERROR ROUTINE
030306  104401  001177                  TYPE    ,$CRLF
030312                          2$:
030312  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
030320  001007                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
030322  113737  001116  030334          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
030330  004737  031152                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
030334     000                  21$:    .BYTE   0
030335     000                  .BYTE   0
030336  000777                  22$:    BR      22$             ;;APT ERROR LOOP
030340  005777  150600          2$:     TST     @SWR            ;;HALT ON ERROR
030344  100001                  BPL     3$              ;;SKIP IF CONTINUE
030346  000000                  HALT                    ;;HALT ON ERROR!
030350  032777  001000  150566  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
030356  001402                  BEQ     4$              ;;BR IF NO
030360  013716  001112                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
030364  005737  001170          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
030370  001402                  BEQ     5$              ;;BR IF NONE
030372  013716  001170                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
030376                          5$:
```



```
030376  022737  022106  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
030404  001001                  BNE     6$              ;;BRANCH IF NO
030406  000000                  HALT                    ;;YES
030410                          6$:
030412  000002                  5$:     RTI                     ;RETURN
```

```
                                ;;*****************************************************************
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE (MODIFIED SYSMAC)

                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (@ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED.  IT THEN OBTAINS, FROM THE "ERROR TABLE",
                                ;*(@ERRTB) THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES TO PRINT.
                                ;*      FORMAT:         W1:     PTR TO ASCIZ ERROR MESSAGE, 0 IF NONE
                                ;*                      W2:     PTR TO ASCIZ DATA HEADER, 0 IF NONE
                                ;*                      W3:     PTR TO DATA VALUES ADDR LIST, 0 IF NONE
                                ;*                      W4-W9:  PTR TO OPERAND VALUES ADDR LIST, 0 IF NONE
                                ;*                      W10:    ALWAYS 0
                                ;*
                                ;*                      DATA VALUES LIST FORMAT:
                                ;*                              A VARIABLE LENGTH LIST OF POINTERS TO
                                ;*                              WORDS TO PRINT AS 6 OCTAL DIGITS.  LIST
                                ;*                              MUST BE TERMINATED BY A ZERO WORD.
                                ;*
                                ;*                      OPERAND VALUES LIST FORMAT:
                                ;*                              FIRST WORD IS ADDRESS OF ASCIZ MESSAGE TO
                                ;*                              PRINT AT START OF LINE; REST OF LIST IS
                                ;*                              IN SAME FORMAT AS DATA VALUES LIST

030412                          $TYPERR:
030412  104401                          TYPE                    ;START WITH MESSAGE PREFIX, HOT OR WARM
030414  001177                  HOTWRN: .WORD   $CRLF            ;PTR TO "HOT" OR "WARM"
030416  010046                          MOV     R0,-(SP)        ;SAVE R0
030420  010146                          MOV     R1,-(SP)        ;SAVE R1
030422  005000                          CLR     R0              ;PICKUP ITEM INDEX
030424  153700  001116                  BISB    @#$ITEMB,R0     ;
030430  001004                          BNE     1$              ;IF ITEM NUMBER FROM ERROR 0,
                                        ;  JUST TYPE PC OF ERROR
030432  013746  001122                  MOV     $ERRPC,-(SP)    ;GET ERROR PC FOR TYPEOUT
030436  104402                          TYPOC                   ;TYPE OCTAL, ALL DIGITS
030440  000454                          BR      7$              ;EXIT
030442  005300                  1$:     DEC     R0              ;ADJUST ERROR # FOR TABLE INDEX
030444  006300                          ASL     R0              ;  OF 20. BYTES/ENTRY
030446  006300                          ASL     R0              ;
030450  010001                          MOV     R0,R1           ;
030452  006300                          ASL     R0              ;
030454  006300                          ASL     R0              ;
030456  060100                          ADD     R1,R0           ;
030460  062700  001232                  ADD     #$ERRTB,R0      ;FORM TABLE PTR
030464  012037  030476                  MOV     (R0)+,2$        ;PICKUP "ERROR MESSAGE" PTR
030470  001404                          BEQ     3$              ;SKIP TYPEOUT IF NULL
030472  104401                          TYPE                    ;TYPE "ERROR MESSAGE"
030474  000000                  2$:     .WORD   0               ;"ERROR MESSAGE" PTR HERE
030476  104401  001177                  TYPE    ,$CRLF          ;CR & LF
030502  104401  030632          3$:     TYPE    ,11$            ;"TEST #    ERR PC" HEADER
030506  012037  030516                  MOV     (R0)+,4$        ;PICKUP "DATA HEADER" PTR
030512  001402                          BEQ     5$              ;SKIP TYPEOUT IF NULL
030514  104401                          TYPE                    ;TYPE "DATA HEADER"
030516  000000                  4$:     .WORD   0               ;"DATA HEADER" PTR HERE
030520  104401  001177          5$:     TYPE    ,$CRLF          ;CR & LF
030524  017746  000074                  MOV     @#$,-(SP)       ;($TESTN)
```



```
030530  104402                          TYPOC                   ;OCTAL W/ LEADING ZEROS
030532  104401  030630                  TYPE    ,10$            ;<HT>
030536  017746  000064                  MOV     @#$,-(SP)       ;($ERRPC)
030542  104402                          TYPOC                   ;OCTAL W/ LEADING ZEROS
030544  104401  030630                  TYPE    ,10$            ;<HT>
030550  012001                          MOV     (R0)+,R1        ;PICKUP "DATA TABLE" PTR
030552  001407                          BEQ     7$              ;EXIT IF NULL
030554  013146                  6$:     MOV     @(R1)+,-(SP)    ;SAVE ... FOR TYPEOUT
030556  104402                          TYPOC                   ;TYPE OCTAL, ALL DIGITS
030560  005711                  14$:    TST     (R1)            ;ANOTHER NUMBER ?
030562  001403                          BEQ     7$              ;NO - EXIT
030564  104401  030630                  TYPE    ,10$            ;TAB BETWEEN ELEMENTS
030570  000771                          BR      6$              ;LOOP ON DATA TABLE VECTOR
030572  104401  001177          7$:     TYPE    ,$CRLF          ;END THE LINE
030576  012001                          MOV     (R0)+,R1        ;GET OPERAND LIST PTR
030600  001406                          BEQ     15$             ;ALL DONE
030602  012137  030612                  MOV     (R1)+,12$       ;OPERAND TITLE
030606  001402                          BEQ     13$             ;SKIP IF ZERO
030610  104401                          TYPE                    ;TYPE IT
030612  000000                  12$:    .WORD   0               ;FROM HERE
030614  000761                  13$:    BR      14$             ;GO FINISH DATA LIST
030616  012601                  15$:    MOV     (SP)+,R1        ;RESTORE R1
030620  012600                          MOV     (SP)+,R0        ;RESTORE R0
030622  000207                          RTS     PC              ;RETURN
030624  001206                  8$:     .WORD   $TESTN          ;
030626  001122                  9$:     .WORD   $ERRPC          ;
030630  000011                  10$:    .ASCIZ  <11>            ;<HT>
030632  042524  052123  021440  11$:    .ASCIZ  "TEST # ERR PC  "
030640  042411  051122  050040
030646  004503      000
030652                                  .EVEN
```

```
                                .SBTTL  TYPE ROUTINE

                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
                                
004652  105737  001165          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
004656  100002                          BPL     1$              ;;BR IF YES
004660  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
004662  000430                          BR      3$              ;;LEAVE
004664  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
004666  017600  000002                  MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
004672  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
005000  001011                          BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
005002  132737  000100  001223          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
005010  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
005012  010037  005722                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
005016  004737  031142                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
005022  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
005024  132737  000040  001223  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
005032  001003                          BNE     6$              ;;YES,SKIP TYPE OUT
005034  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
005036  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
005040  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
005042  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
005044  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
005050  000002                          RTI                     ;;RETURN
005052  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
005056  001430                          BEQ     8$
005060  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
005064  001006                          BNE     5$
005066  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
005070  104401                          TYPE                    ;;TYPE A CR AND LF
005072  031177                          $CRLF
005074  105037  031130                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
005100  000755                          BR      2$              ;;GET NEXT CHARACTER
005102  004737  031064          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
005106  123726  001164          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
005112  001350                          BNE     2$              ;;IF NO GO GET NEXT*CHAR.
005114  013746  001162                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
005120  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
005124  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
005126  004737  031064                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
005132  105337  031130                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```



```
005136  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

005140  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
005144  004737  031064          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
005150  132737  000007  031130          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
005156  001372                          BNE     9$              ;;TAB STOP
005160  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
005162  000724                          BR      2$              ;;GET NEXT CHARACTER
005164  105777  150066          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
005170  100375                          BPL     $TYPEC
005172  116677  000002  150060          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
005200  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
005206  001001                          BNE     1$              ;;BRANCH IF NO
005210  105037  031130                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
005214  000406                          BR      $TYPEX          ;;EXIT
005216  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
005224  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
005226  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
005230  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
005232  000207          $TYPEX: RTS     PC
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;*****************************************************************
031134  112737  000001  031404  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
031142  112737  000001  031376  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
                                        BR      $ATYC
031150  000403
031152  112737  000001  031400  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
031160                          $ATYC:
031160  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
031162  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
031164  105737  031376                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
031170  001450                          BEQ     5$              ;;IF NOT:  BR
031172  122737  000001  001222          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
031200  001031                          BNE     3$              ;;IF NOT:  BR
031202  132737  000100  001223          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
031210  001425                          BEQ     3$              ;;IF NOT:  BR
031212  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
031216  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
031224  005737  001202          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
031230  001375                          BNE     1$              ;;IF NOT:  WAIT
031232  010037  001216                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
031236  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
031240  001376                          BNE     2$
031242  163700  001216                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
031246  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
031250  010037  001220                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
031254  012737  000004  001202          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
031262  000413                          BR      5$
031264  017637  000004  031310  3$:     MOV     @4(SP),4$               ;;PUT MSG ADDR IN JSR LINKAGE
031272  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
031300  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
031304  004737  030652                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
031310  000000                  4$:     .WORD   0
                                5$:
031312                          1P$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
031312  105737  031400                  BEQ     12$             ;;IF NOT:  BR
031316  001416                          TST     $ENV            ;;RUNNING UNDER APT?
031320  005737  001222                  BEQ     12$             ;;IF NOT:  BR
031324  001413                  11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
031326  005737  001202                  BNE     11$             ;;IF NOT:  WAIT
031332  001375                          MOV     @4(SP),$FATAL   ;;GET ERROR #
031334  017637  000004  001204          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
031342  062766  000002  000004          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
031350  005237  001202          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
031354  105037  031400                  CLRB    $LFLG           ;;CLEAR LOG FLAG
031360  105037  031377                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
031364  105037  031376                  MOV     (SP)+,R1        ;;POP STACK INTO R1
031370  012601                          MOV     (SP)+,R0        ;;POP STACK INTO R0
031372  012600                          RTS     PC              ;;RETURN
031374  000207
031376      000         $MFLG:  .BYTE   0               ;;MESSG. FLAG
031377      000         $LFLG:  .BYTE   0               ;;LOG FLAG
031400      000         $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
            031402      APT&IZE=240
            000200      APTENV=001
            000001      APTSPOOL=100
            000100
```



```
            000040      APTCSUP=040
```

```
                                        .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

031402  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
031406  116637  000001  031625          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
031414  112637  031627                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
031420  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
031424  000406                          BR      $TYPON
031426  112737  000001  031625  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
031434  112737  000006  031627          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
031442  112737  000005  031624  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
031450  010346                          MOV     R3,-(SP)        ;;SAVE R3
031452  010446                          MOV     R4,-(SP)        ;;SAVE R4
031454  010546                          MOV     R5,-(SP)        ;;SAVE R5
031456  113704  031627                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
031462  005404                          NEG     R4
031464  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
031470  110437  031626                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
031474  113704  031625                  MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
031500  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
031504  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
031506  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
031510  103404                          BR      3$              ;;GO DO MSB
031512  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
031514  006105                          ROL     R5
031516  006105                          ROL     R5
031520  010503                          MOV     R5,R3
031522  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
031524  105337  031626                  DECB    $OMODE          ;;TYPE THIS DIGIT?
031530  100016                          BPL     7$              ;;BR IF NO
031532  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
031536  001002                          BNE     4$              ;;TEST FOR 0
031540  005704                          TST     R4              ;;SUPPRESS THIS 0?
031542  001403                          BEQ     5$              ;;BR IF YES
```



```
031544  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
031546  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
031552  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
031556  110337  031622                  MOVB    R3,$S           ;;SAVE FOR TYPING
031562  104401  031622                  TYPE    ,B$             ;;GO TYPE THIS DIGIT
031566  105337  031624          7$:     DECB    $OCNT           ;;COUNT BY 1
031572  003347                          BGT     2$              ;;BR IF MORE TO DO
031574  002402                          BLT     6$              ;;BR IF DONE
031576  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
031600  000744                          BR      2$              ;;GO DO THE LAST DIGIT
031602  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
031604  012604                          MOV     (SP)+,R4        ;;RESTURE R4
031606  012603                          MOV     (SP)+,R3        ;;RESTORE R3
031610  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
031616  012616                          MOV     (SP)+,(SP)
031620  000002                          RTI                     ;;RETURN
031622      000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
031623      000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
031624      000         $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
031625      000         $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
031626  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .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.

031630  012046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
031632  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
031636  005740                  TST     -(R0)           ;;BACKUP BY 2
031640  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
031642  006300                  ASL     R0              ;;POSITION FOR INDEXING
031644  016000  031664          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
031650  000200                  RTS     R0              ;;GO TO ROUTINE


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

031652  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
031654  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
031662  000002                  RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
031664  031652                  $TRPAD: .WORD   $TRAP2
                                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
                                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
                                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
031677  031682                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```



```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*************************************************************
                                ;POWER DOWN ROUTINE
031676  012737  032050  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
031704  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
031712  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
031714  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
031716  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
031720  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
031722  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
031724  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
031726  017746  147212          MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
031732  010637  032054          MOV     SP,$SAVR6       ;;SAVE SP
031736  012737  031750  000024  MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
031744  000000                  HALT
031746  000776                  BR      .-2             ;;HANG UP

                                ;;*************************************************************
                                ;POWER UP ROUTINE
031750  012737  032050  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
031756  013706  032054          MOV     $SAVR6,SP       ;;GET SP
031762  005037  032054          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
031766  005237  032054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
031772  001375                  BNE     1$              ;;OF  WORD
031774  011600                  MOV     (SP),R0         ;GET SAVED SWR OFF STACK
031776  076600  000226          MFD     ,226            ;RESTORE SWR CONTENTS
032002  012677  147136          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
032006  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
032010  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
032012  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
032014  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
032016  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
032020  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
032022  012737  031676  000024  MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
032030  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;PRIO:7
032036  104401                  TYPE                    ;REPORT THE POWER FAILURE
032040  032056          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
032042  012716                  MOV     (PC)+,(SP)      ;;RESTART AT START
032044  003000          $PWRAD: .WORD   START           ;;RESTART ADDRESS
032046  000002                  RTI
032050  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
032052  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
032054  000000          $SAVR6: 0                       ;;PUT THE SP HERE
032056  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
032064  000122

                                .EVEN
```

```
5001                                            .SBTTL  ERROR MESSAGES, DATA HEADERS, DATA VECTORS, OPERAND VECTORS, ETC
5002
5003                                            ;MESSAGE PREFIX
5004  032066  047516  035124  000040  ASCHOT: .ASCIZ  "HOT: "
5005  032074  040527  046522  020072  ASCWRM: .ASCIZ  "WARM: "
5006  032102      000
5007
5008                                            ;ERROR MESSAGES HERE
5009  032103      106  046455  042117  EMA:    .ASCIZ  "F-MODE EXERCISER - FPS ERROR"
5010  032110  020105  054105  051105
5011  032116  044503  042523  020122
5012  032124  020055  050106  020123
5013  032132  051105  047522  000122
5014  032140  026504  047515  042504  EMB:    .ASCIZ  "D-MODE EXERCISER - FPS ERROR"
5015  032146  042440  042530  041522
5016  032154  051511  051105  026440
5017  032162  043040  051520  042440
5018  032170  051122  051117      000
5019  032175      106  046455  042117  EMC:    .ASCIZ  "F-MODE EXERCISER - RESULT ERROR"
5020  032202  020105  054105  051105
5021  032210  044503  042523  020122
5022  032216  020055  042522  052523
5023  032224  052114  042440  051122
5024  032232  051117      000
5025  032235      104  046455  042117  EMD:    .ASCIZ  "D-MODE EXERCISER - RESULT ERROR"
5026  032242  020105  054105  051105
5027  032250  044503  042523  020122
5028  032256  020055  042522  052523
5029  032264  052114  042440  051122
5030  032272  051117      000
5031  032275      101  042104  043050  EME:    .ASCIZ  "ADD(F/D) - RESULT ERROR"
5032  032302  042057  020051  020055
5033  032310  042522  052523  052114
5034  032316  042440  051122  051117
5035  032324      000
5036  032325      123  041125  043050  EMF:    .ASCIZ  "SUB(F/D) - RESULT ERROR"
5037  032332  042057  020051  020055
5038  032340  042522  052523  052114
5039  032346  042440  051122  051117
5040  032354      000
5041  032355      115  046125  043050  EMG:    .ASCIZ  "MUL(F/D) - RESULT ERROR"
5042  032362  042057  020051  020055
5043  032370  042522  052523  052114
5044  032376  042440  051122  051117
5045  032404      000
5046  032405      104  053111  043050  EMH:    .ASCIZ  "DIV(F/D) - RESULT ERROR"
5047  032412  042057  020051  020055
5048  032420  042522  052523  052114
5049  032426  042440  051122  051117
5050  032434      000
5051  032435      125  042516  050130  EMI:    .ASCIZ  "UNEXPECTED FLOATING POINT TRAP, IGNORED AND CONTINUING"
5052  032442  041505  042524  020104
5053  032450  046106  040517  044524
5054  032456  043516  050040  044517
5055  032464  052116  052040  040522
5056  032472  026120  044440  047107
```



```
5057  032500  051117  042105  040440
5058  032506  042116  041440  047117
5059  032514  044524  052516  047111
5060  032522  000107
5061  032524  042101  043104  026440  EMJ:    .ASCIZ  "ADDF - FPS ERROR"
5062  032532  043040  051520  042440
5063  032540  051122  051117      000
5064  032545      123  041125  020106  EMK:    .ASCIZ  "SUBF - FPS ERROR"
5065  032552  020055  050106  020123
5066  032560  051105  047522  000122
5067  032566  052515  043114  026440  EML:    .ASCIZ  "MULF - FPS ERROR"
5068  032574  043040  051520  042440
5069  032602  051122  051117      000
5070  032607      104  053111  020106  EMM:    .ASCIZ  "DIVF - FPS ERROR"
5071  032614  020055  050106  020123
5072  032622  051105  047522  000122
5073  032630  042101  042104  026440  EMN:    .ASCIZ  "ADDD - FPS ERROR"
5074  032636  043040  051520  042440
5075  032644  051122  051117      000
5076  032651      123  041125  020104  EMO:    .ASCIZ  "SUBD - FPS ERROR"
5077  032656  020055  050106  020123
5078  032664  051105  047522  000122
5079  032672  052515  042114  026440  EMP:    .ASCIZ  "MULD - FPS ERROR"
5080  032700  043040  051520  042440
5081  032706  051122  051117      000
5082  032713      104  053111  020104  EMQ:    .ASCIZ  "DIVD - FPS ERROR"
5083  032720  020055  050106  020123
5084  032726  051105  047522  000122
5085  032734  042101  043104  026440  EMR:    .ASCIZ  "ADDF - FEC/FEA ERROR"
5086  032742  043040  041505  043057
5087  032750  040505  042440  051122
5088  032756  051117      000
5089  032761      123  041125  020106  EMS:    .ASCIZ  "SUBF - FEC/FEA ERROR"
5090  032766  020055  042506  027503
5091  032774  042506  020101  051105
5092  033002  047522  000122
5093  033006  052515  043114  026440  EMT:    .ASCIZ  "MULF - FEC/FEA ERROR"
5094  033014  043040  041505  043057
5095  033022  040505  042440  051122
5096  033030  051117      000
5097  033033      104  053111  020106  EMU:    .ASCIZ  "DIVF - FEC/FEA ERROR"
5098  033040  020055  042506  027503
5099  033046  042506  020101  051105
5100  033054  047522  000122
5101  033060  042101  042104  026440  EMV:    .ASCIZ  "ADDD - FEC/FEA ERROR"
5102  033066  043040  041505  043057
5103  033074  040505  042440  051122
5104  033102  051117      000
5105  033105      123  041125  020104  EMW:    .ASCIZ  "SUBD - FEC/FEA ERROR"
5106  033112  020055  042506  027503
5107  033120  042506  020101  051105
5108  033126  047522  000122
5109  033132  052515  042114  026440  EMX:    .ASCIZ  "MULD - FEC/FEA ERROR"
5110  033140  043040  041505  043057
5111  033146  040505  042440  051122
5112  033154  051117      000
```

```
5113  033157    104  053111  020104  ENV:    .ASCIZ  "DIVD - FEC/FEA ERROR"
5114  033164  028055  042506  027503
5115  033172  042506  020101  051105
5116  033200  047522  000122
5117
5118
5119                                          ;DATA HEADERS HERE
5120  033204  054105  023520  004504  DHA:    .ASCIZ  "EXP'D  RCV'D"
5121  033212  041522  023526  000104
5122  033220  026455  002455  050130  DHB:    .ASCIZ  "---EXPECTED--- ---RECEIVED---"
5123  033226  041505  042524  026504
5124  033234  026455  026411  026455
5125  033242  042522  042503  053111
5126  033250  042105  026455  000055
5127  033256  026455  026455  026455  DHC:    .ASCIZ  "-----------EXPECTED----------- -----------RECEIVED-----------"
5128  033264  026455  026455  042455
5129  033272  050130  041505  042524
5130  033300  026504  026455  026455
5131  033306  026455  026455  026455
5132  033314  026411  026455  026455
5133  033322  026455  026455  026455
5134  033330  042522  042503  053111
5135  033336  042105  026455  026455
5136  033344  026455  026455  026455
5137  033352  000055
5138  033354  046117  020104  041520  DHD:    .ASCIZ  "OLD PC OLD PS   FPS     FEC     FEA     $FPS    $FEC    $FEA"
5139  033362  047411  042114  050040
5140  033370  004523  043040  051520
5141  033376  020011  042506  004503
5142  033404  043040  040505  020011
5143  033412  043044  051520  020011
5144  033420  043044  041505  020011
5145  033426  043044  040505     000
5146  033433     105  050130  042047  DHE:    .ASCIZ  "EXP'D-FEC-RCV'D         EXP'D-FEA-RCV'D"
5147  033440  043055  041505  051055
5148  033446  053103  042047  042411
5149  033454  050130  042047  043055
5150  033462  040505  051055  053103
5151  033470  042047     000
5152
5153
5154                                          ;DATA VECTORS HERE
5155        033474                            .EVEN
5156  033474  002400  002362  000000  DTA:    .WORD   $FPS,FPS,0
5157  033502  002402  002364  002404  DTB:    .WORD   $FEC,FEC,$FEA,FEA,0
5158  033510  002366  000000
5159  033514  002416  002420  002406  DTD:    .WORD   ANS2,ANS2+2,ANS1,ANS1+2,0
5160  033522  002410  000000
5161  033526  002416  002420  002422  DTE:    .WORD   ANS2,ANS2+2,ANS2+4,ANS2+6
5162  033534  002424
5163  033536  002406  002410  002412          .WORD   ANS1,ANS1+2,ANS1+4,ANS1+6,0
5164  033544  002414  000000
5165  033550  002370  002372  002362  DTF:    .WORD   FPPOPC,FPPOPS,FPS,FEC,FEA,$FPS,$FEC,$FEA,0
5166  033556  002364  002366  002400
5167  033564  002402  002404  000000
5168
```



```
5169                                          ;OPERAND VECTORS HERE
5170                                          .EVEN
5171  033572  034032  002426  002430  LOD:    .WORD   XLO,LONUM,LONUM+2,LONUM+4,LONUM+6,0
5172  033600  002432  002434  000000
5173  033606  034042  002436  002440  HID:    .WORD   XHI,HINUM,HINUM+2,HINUM+4,HINUM+6,0
5174  033614  002442  002444  000000
5175  033622  034032  002426  002430  LOF:    .WORD   XLO,LONUM,LONUM+2,0
5176  033630  000000
5177  033632  034042  002436  002440  HIF:    .WORD   XHI,HINUM,HINUM+2,0
5178  033640  000000
5179  033642  034052  002446  002450  OP1F:   .WORD   XOP1,OP1,OP1+2,0
5180  033650  000000
5181  033652  034060  002456  002460  OP2F:   .WORD   XOP2,OP2,OP2+2,0
5182  033660  000000
5183  033662  034066  002466  002470  OP3F:   .WORD   XOP3,OP3,OP3+2,0
5184  033670  000000
5185  033672  034074  002476  002500  OP4F:   .WORD   XOP4,OP4,OP4+2,0
5186  033700  000000
5187  033702  034110  002506  002510  OP5F:   .WORD   XOP5,OP5,OP5+2,0
5188  033710  000000
5189  033712  034102  002516  002520  OP6F:   .WORD   XOP6,OP6,OP6+2,0
5190  033720  000000
5191  033722  034052  002446  002450  OP1D:   .WORD   XOP1,OP1,OP1+2,OP1+4,OP1+6,0
5192  033730  002452  002454  000000
5193  033736  034060  002456  002460  OP2D:   .WORD   XOP2,OP2,OP2+2,OP2+4,OP2+6,0
5194  033744  002462  002464  000000
5195  033752  034066  002466  002470  OP3D:   .WORD   XOP3,OP3,OP3+2,OP3+4,OP3+6,0
5196  033760  002472  002474  000000
5197  033766  034074  002476  002500  OP4D:   .WORD   XOP4,OP4,OP4+2,OP4+4,OP4+6,0
5198  033774  002502  002504  000000
5199  034002  034110  002506  002510  OP5D:   .WORD   XOP5,OP5,OP5+2,OP5+4,OP5+6,0
5200  034010  002512  002514  000000
5201  034016  034102  002516  002520  OP6D:   .WORD   XOP6,OP6,OP6+2,OP6+4,OP6+6,0
5202  034024  002522  002524  000000
5203
5204
5205                                          ;OPERAND TITLES
5206  034032  047514  052516  035115  XLO:    .ASCIZ  "LONUM:"<11>
5207  034040  000011
5208  034042  044510  052516  035115  XHI:    .ASCIZ  "HINUM:"<11>
5209  034050  000011
5210  034052  050117  035061  000011  XOP1:   .ASCIZ  "OP1:"<11>
5211  034060  050117  035062  000011  XOP2:   .ASCIZ  "OP2:"<11>
5212  034066  050117  035063  000011  XOP3:   .ASCIZ  "OP3:"<11>
5213  034074  050117  035064  000011  XOP4:   .ASCIZ  "OP4:"<11>
5214  034102  050117  035066  000011  XOP6:   .ASCIZ  "OP6:"<11>
5215  034110  050117  035065  000011  XOP5:   .ASCIZ  "OP5:"<11>
5216                                          ;THE END
5217        000001                            .END
```

```
ABASE = 000000            258
ACDW1 = 000000            258
ACDW2 = 000000            258
ACPUOP= 000000            258     273
ADDC0   002526           462#     570    3372    3671*
ADDC1   002530           463#    3372    3715*
ADDC2   002532           464#    3372    3694*
ADDC3   002534           465#    3372    3717*
ADDC4   002536           466#    3372    3760*
ADDC5   002540           467#    3372    3821*   3841*
ADDC6   002542           468#    3372    3824*   3844*
ADDC7   002544           469#    3372    3855*   3901*
ADDC8   002546           470#    3372    3902*
ADDWM = 000000            258
ADDW1 = 000000            258
ADDW10= 000000            258
ADDW11= 000000            258
ADDW12= 000000            258
ADDW13= 000000            258
ADDW14= 000000            258
ADDW15= 000000            258
ADDW2 = 000000            258
ADDW3 = 000000            258
ADDW4 = 000000            258
ADDW5 = 000000            258
ADDW6 = 000000            258
ADDW7 = 000000            258
ADDW8 = 000000            258
ADDW9 = 000000            258
ADEVCT= 000000            258     264
ADEVM = 000000            258
AEND1   004102           706     713    717#
AEND10  006456          1151    1160    1167    1171*
AEND11  006742          1224    1231    1235#
AEND12  007236          1291    1300    1304#
AEND13  007464          1348    1355    1359#
AEND14  007722          1403    1412    1416#
AEND15  010230          1472    1479    1484    1488#
AEND16  010556          1544    1553    1560    1564#
AEND17  011026          1600    1615    1620    1624#
AEND2   004376           772     781     785#
AEND20  011316          1660    1677    1684    1688#
AEND3   004624           829     836     840#
AEND4   005062           884     893     897#
AEND5   005370           953     960     965     969#
AEND6   005716          1026    1035    1042    1046#
AEND7   006166          1091    1098    1103    1107#
AENV  = 000000            258     269
AENVM = 000000            258     270
AERR1   004002           688     692#
AERR11  006642          1206    1210#
AERR12  007126          1273    1277#
AERR15  010106          1454    1458#
AERR16  010414          1526    1530#
AERR2   004266           754     758#
AERR5   005246           935     939#
```



```
AERR6   005554          1008    1012#
AFATAL= 000000            258     261
AMADR1= 000000            258
AMADR2= 000000            258
AMADR3= 000000            258
AMADR4= 000000            258
AMAMS1= 000000            258
AMAMS2= 000000            258
AMAMS3= 000000            258
AMAMS4= 000000            258
AMSGAD= 000000            258     266
AMSGLG= 000000            258     267
AMSGTY= 000000            258     260
AMTYP1= 000000            258
AMTYP2= 000000            258
AMTYP3= 000000            258
AMTYP4= 000000            258
ANS1    002406           440#    689*    708*    709*    710*    711     714*    755*    774*    775*    776*    777*    778*
                         779     782*    824*    831*    832*    833*    834     837*    879*    886*    887*    888*    889*
                         890*    891     894*    936*    955*    956*    957*    958     961*    962*    963     966*    1009*
                        1028*   1029*   1030*   1031*   1032*   1033    1036*   1037*   1038*   1039*   1040    1043*   1086*
                        1093*   1094*   1095*   1096    1099*   1100*   1101    1104*   1146*   1153*   1154*   1155*   1156*
                        1157*   1158    1161*   1162*   1163*   1164*   1165    1168*   1207*   1226*   1227*   1228*   1229
                        1232*   1274*   1293*   1294*   1295*   1296*   1297*   1298    1301*   1343*   1350*   1351*   1352*
                        1353    1356*   1398*   1405*   1406*   1407*   1408*   1409*   1410    1413*   1455*   1474*   1475*
                        1476*   1477    1480*   1481*   1482    1485*   1527*   1546*   1547*   1548*   1549*   1550*   1551
                        1554*   1555*   1556*   1557*   1558    1561*   1603*   1610*   1611*   1612*   1613    1616*   1617*
                        1618    1671*   1663*   1670*   1671*   1672*   1673*   1674*   1675    1678*   1679*   1680*   1681*
                        1682    1685*   1724*   1743*   1744*   1745*   1746    1749*   1791*   1810*   1811*   1812*   1813*
                        1814*   1815    1818*   1860*   1867*   1868*   1869*   1870    1873*   1915*   1922*   1923*   1924*
                        1925*   1926*   1927    1930*   1971*   1990*   1991*   1992*   1993    1996*   1997*   1998    2001*
                        2043*   2062*   2063*   2064*   2065*   2066*   2067    2070*   2071*   2072*   2073*   2074    2077*
                        2119*   2126*   2127*   2128*   2129    2132*   2133*   2134    2137*   2179*   2186*   2187*   2188*
                        2189*   2190*   2191    2194*   2195*   2196*   2197*   2198    2201*   2240*   2259*   2260*   2261*
                        2262    2265*   2307*   2326*   2327*   2328*   2329*   2330*   2331    2334*   2376*   2395*   2396*
                        2397*   2398    2401*   2443*   2462*   2463*   2464*   2465*   2466*   2467    2470*   2512*   2531*
                        2532*   2533*   2534    2537*   2538*   2539    2542*   2584*   2603*   2604*   2605*   2606*   2607*
                        2608    2611*   2612*   2613*   2614*   2615    2618*   2660*   2679*   2680*   2681*   2682    2685*
                        2686*   2687    2690*   2732*   2751*   2752*   2753*   2754*   2755*   2756    2759*   2760*   2761*
                        2762*   2763    2766*   2808*   2827*   2828*   2829*   2830    2833*   2874*   2893*   2894*   2895*
                        2896*   2897*   2898    2901*   2943*   2962*   2963*   2964*   2965    2968*   2969*   2970    2973*
                        3014*   3033*   3034*   3035*   3036*   3037*   3038    3041*   3042*   3043*   3044*   3045    3048*
                        3107*   3115*   3117*   3118*   3119*   3121*   3122*   3123*   3124*   3126    3129    3132*   3197*
                        3205*   3207*   3208*   3209*   3210*   3211*   3212*   3213*   3214*   3215*   3216*   3218*   3219*
                        3220*   3221*   3222*   3223*   3224*   3225*   3227    3229    3231    3234    3237*   5159    5163
ANS2    002416           442#    672     678     738     744     807     813     862     868     919     925     992     998
                        1069    1075    1129    1135    1190    1196    1257    1263    1326    1332    1381    1387    1438
                        1444    1510    1516    1586    1592    1646    1652    1707    1713    1774    1780    1843    1849
                        1898    1904    1954    1960    2026    2032    2102    2108    2162    2168    2223    2229    2290
                        2296    2359    2365    2426    2432    2495    2501    2567    2573    2643    2649    2715    2721
                        2791    2797    2857    2863    2926    2932    2997    3003    3006    3089    3117    3119    3176
                        3179    3207    3211    3214    3216    5159    5161
APAS5 * 000000            258     263
APRIOR= 000000            258
APTCSU= 000046           4734    4839#
APTENV= 000001           4595    4727    4795    4837#
```

```
APTS12= 000200            560    4836#
APTSPC= 000140           4729    4797    4838#
ARET1   003756            659    685#
ARET10  006326           1116    1142#
ARET11  006616           1177    1203#
ARET12  007102           1244    1270#
ARET13  007376           1313    1339#
ARET14  007624           1360    1394#
ARET15  010062           1425    1451#
ARET16  010370           1497    1523#
ARET17  010716           1573    1599#
ARET2   004242            725    751#
ARET20  011166           1633    1659#
ARET3   004536            794    820#
ARET4   004764            849    875#
ARET5   005222            906    932#
ARET6   005530            979    1005#
ARET7   006056           1056    1082#
A6CHOT  032066            651    5004#
A6CWRM  032074            653    5005#
A6WREG= 000000            258    271
ATE6TN= 000000            258    262
ATST1   004040            693    701     704#
ATST10  006352           1145    1149#
ATST11  006700           1211    1219    1222#
ATST12  007164           1278    1286    1289#
ATST13  007422           1342    1346#
ATST14  007650           1397    1401#
ATST15  010144           1459    1467    1470#
ATST16  010452           1531    1539    1542#
ATST17  010742           1602    1606#
ATST2   004324            759    767     770#
ATST20  011212           1662    1666#
ATST3   004562            823    827#
ATST4   005010            870    882#
ATST5   005304            940    948     951#
ATST6   005612           1013    1021    1024#
ATST7   006102           1085    1089#
AUNIT = 000004            258    265
AUSWR = 000000            258    272
AVECT1= 000000            256
AVECT2= 000000            258
BGNMES  002622            498#    575
BIT0  = 000001            125#
BIT00 = 000001            115#    125
BIT01 = 000002            114#    124
BIT02 = 000004            113#    123
BIT03 = 000010            112#    122
BIT04 = 000020            111#    121     581     633     3260
BIT05 = 000040            110#    120
BIT06 = 000100            109#    119
BIT07 = 000200            108#    118
BIT08 = 000400            107#    117     4522
BIT09 = 001000            106#    116     4530    4605
BIT1  = 000002            124#
BIT10 = 002000            105#    4583
```



```
BIT11 = 004000            104#    4537
BIT12 = 010000            103#    613     643     645     648     3266
BIT13 = 020000            102#    4590
BIT14 = 040000            101#    4508
BIT15 = 100000            100#
BIT2  = 000004            123#
BIT3  = 000010            122#
BIT4  = 000020            121#
BIT5  = 000040            120#
BIT6  = 000100            119#
BIT7  = 000200            118#
BIT8  = 000400            117#
BIT9  = 001000            116#
BPTVEC= 000014            132#
CR    = 000015            488#    498     503     511     4773    4783
CRLF  = 000200            418#    4744    4783
DDISP = 177570            478#    238     548
DEND1   014442           2257    2264    2268#
DEND10  017206           2744    2749    2758    2765    2769#
DEND11  017472           2825    2832    2836#
DEND12  017766           2891    2900    2904#
DEND13  020274           2960    2967    2972    2976#
DEND14  020622           3031    3040    3047    3051#
DEND2   014736           2324    2333    2337#
DEND3   015222           2393    2400    2404#
DEND4   015516           2460    2469    2473#
DEND5   016024           2529    2536    2541    2545#
DEND6   016352           2601    2610    2617    2621#
DEND7   016660           2677    2684    2689    2693#
DERR1   014342           2239    2243#
DERR10  017044           2731    2735#
DERR11  017372           2807    2811#
DERR12  017656           2873    2877#
DERR13  020152           2942    2946#
DERR14  020460           3013    3017#
DERR2   014626           2306    2310#
DERR3   015122           2375    2379#
DERR4   015406           2442    2446#
DERR5   015702           2511    2515#
DERR6   016210           2583    2587#
DERR7   016536           2659    2663#
DHA     033204            298    302     306     354     358     362     366     378     408     412     5120#
DHB     033220            316    324     332     340     417     5122#
DHC     033256            320    328     336     344     421     5127#
DHD     033354            349    5138#
DHE     033433            375    379     383     387     391     395     399     403     5146#
DISPLA  001146            238#    548#    556#    4552#   4582#
DISPRE  000174            169#    556
DIVC0   002564            479#    3379    4263#
DIVC1   002566            480#    3379    4291#
DIVC2   002570            481#    3379    4276#
DIVC3   002572            482#    3379    4378#   4399#
DIVC4   002574            483#    3379    4381#   4402#
DIVC5   002576            484#    3379    4367#
DIVC6   002600            485#    3379    4370#
DRET1   014316           2210    2236#
```

```
DRET10  #17020          2702    2728#
DRET11  #17346          2778    2804#
DRET12  #17632          2844    2870#
DRET13  #20126          2913    2939#
DRET14  #20434          2984    3010#
DRET2   #14602          2277    2303#
DRET3   #15076          2346    2372#
DRET4   #15362          2413    2439#
DRET5   #15656          2482    2508#
DRET6   #16164          2554    2580#
DRET7   #16512          2630    2656#
DSWR  = 17757#            46#    237     547
DTA     #33474           298     302     306     354     358     362     366     370     400     412    5156#
DTB     #33502           375     379     383     397     391     395     399     403    5157#
DTD     #33514           316     324     332     340     417    5159#
DTE     #33526           320     328     336     344     421    5161#
DTF     #33550           349    5165#
DTST1   #14400          2244    2252    2258#
DTST10  #17102          2736    2747#
DTST11  #17430          2812    2820    2823#
DTST12  #17714          2878    2886    2889#
DTST13  #20210          2947    2955    2958#
DTST14  #20516          3018    3026    3029#
DTST2   #14664          2311    2319    2322#
DTST3   #15160          2380    2388    2391#
DTST4   #15444          2447    2455    2458#
DTST5   #15740          2516    2524    2527#
DTST6   #16246          2588    2596    2599#
DTST7   #16574          2664    2672    2675#
EEND1   #21200          3112    3130    3135#
EEND2   #21662          3202    3235    3240#
EERR1   #21172          3127    3132#
EERR2   #21654          3229    3230    3232    3237#
EMA     #32103           408    5009#
EMB     #32140           412    5014#
EMC     #32175           417    5019#
EMD     #32235           421    5025#
EME     #32275           316     320    5031#
EMF     #32325           324     328    5036#
EMG     #32355           332     336    5041#
EMH     #32405           340     344    5046#
EMI     #32435           349    5051#
EMJ     #32524           298    5061#
EMK     #32545           302    5064#
EML     #32566           306    5067#
EMM     #32607           354    5070#
EMN     #32630           358    5073#
EMO     #32651           362    5076#
EMP     #32672           366    5079#
EMQ     #32713           370    5082#
EMR     #32734           375    5085#
EMS     #32761           379    5089#
EMT     #33006           383    5093#
EMTVEC= #000030          135#    531*    532*
EMU     #33031           387    5097#
EMV     #33060           391    5101#
```



```
EMV001  #01232           298#
EMV002  #01256           302#
EMV003  #01302           306#
EMV004  #01326           311#
EMV005  #01352           316#
EMV006  #01376           320#
EMV007  #01422           324#
EMV010  #01446           328#
EMV011  #01472           332#
EMV012  #01516           336#
EMV013  #01542           340#
EMV014  #01566           344#
EMV015  #01612           349#
EMV016  #01636           354#
EMV017  #01662           358#
EMV020  #01706           362#
EMV021  #01732           366#
EMV022  #01756           370#
EMV023  #02002           375#
EMV024  #02026           379#
EMV025  #02052           383#
EMV026  #02076           387#
EMV027  #02122           391#
EMV030  #02146           395#
EMV031  #02172           399#
EMV032  #02216           403#
EMV033  #02242           408#
EMV034  #02266           412#
EMV035  #02312           417#
EMV036  #02336           421#
EMW     #33105           395    5105#
EMX     #33132           399    5109#
EMY     #33157           403    5113#
EREG0   #02602           480#   4571#
EREG1   #02604           489#   4572#
EREG2   #02606           490#   4573#
EREG3   #02610           491#   4574#
EREG4   #02612           492#   4575#
EREG5   #02614           493#   4576#
EREG6   #02616           494#   4577#   4578#
EREG7   #02620           495#   4579#
ERPVEC= #000004          128#    545     546*    557*    4513    4514*   4516*   4519*
ETST1   #21076          3186    3110#
ETST2   #21464          3196    3200#
EXPFEA  #02376           436#    659*    725*    794*    849*    906*    979#   1056*   1116*   1177*   1244*   1313*   1368*
                        1425*   1497*   1573*   1633*   1694*   1761*   1830*   1885*   1941*   2013*   2089*   2149*   2210*
                        2277*   2346*   2413*   2482*   2554*   2630*   2702*   2778*   2844*   2913*   2984*   3027    3847
                        3905    4086    4097    4128    4279    4373    4384    4405
FEA     #02366           432#    665*    700     731*    766     800*    855*    912*    947     985*    1020    1062*   1122*
                        1183*   1218    1250*   1285    1319*   1374*   1431*   1466    1503*   1538    1579*   1639*   1700*
                        1735    1767*   1802    1836*   1891*   1947*   1982    2019*   2054    2095*   2155*   2216*   2251
                        2283*   2318    2352*   2387    2419*   2454    2488*   2523    2560*   2595    2636*   2671    2708*
                        2743    2784*   2819    2850*   2885    2919*   2954    2990*   3025    3063*   3151*   5157    5165
FEC     #02364           431#    664*    694*    696     730*    760*    762     799*    854*    911*    941*    943     984*
                        1014*   1016    1061*   1121*   1182*   1212*   1214    1249*   1279*   1281    1318*   1373*   1430*
                        1460*   1462    1502*   1532*   1534    1578*   1638*   1699*   1729*   1731    1766*   1796*   1798
```

```
                    1835#  1890#  1946#  1976#  1978   2018#  2048#  2050   2094#  2154#  2215#  2245#  2247
                    2282#  2312#  2314   2351#  2381#  2383   2418#  2448#  2450   2487#  2517#  2519   2559#
                    2589#  2591   2635#  2665#  2667   2707#  2737#  2739   2783#  2813#  2815   2849#  2879#
                    2881   2918#  2948#  2950   2989#  3019#  3021   3062#  3150#  3477#  5157   5165
FPPILI   023256      566    3474#
FPPOPC   002370     4338   3474#  3487   5165
FPPOPS   002372     4348   3475#  3486   5165
FPPOSP   002374     4358
FPPVEC=  000244     1508    566#   567#
FPS      002362     4308    663#   686#   687    717#   729#   752#   753    785#   798#   821#   822    840#
                     853#   876#   877    897#   910#   933#   934    969#   983#  1006#  1007   1046#  1060#
                    1083#  1084   1107#  1120#  1143#  1144   1171#  1181#  1204#  1205   1235#  1248#  1271#
                    1272   1304#  1317#  1340#  1341   1359#  1372#  1395#  1396   1416#  1429#  1452#  1453
                    1480#  1501#  1524#  1525   1564#  1577#  1600#  1601   1624#  1637#  1660#  1661   1688#
                    1698#  1721#  1722   1752#  1765#  1788#  1789   1821#  1834#  1857#  1858   1876#  1889#
                    1912#  1913   1933#  1945#  1968#  1969   2004#  2017#  2040#  2041   2080#  2093#  2116#
                    2117   2140#  2153#  2176#  2177   2204#  2214#  2237#  2238   2268#  2281#  2304#  2305
                    2337#  2358#  2373#  2374   2404#  2417#  2440#  2441   2473#  2486#  2509#  2510   2545#
                    2558#  2581#  2582   2621#  2634#  2657#  2658   2693#  2706#  2729#  2730   2769#  2782#
                    2805#  2806   2836#  2848#  2871#  2872   2904#  2917#  2940#  2941   2976#  2988#  3011#
                    3012   3051#  3061#  3104#  3105   3135#  3149#  3194#  3195   3240#  3476#  3478   3481
                    5156   5165
GNS    = ******  U   168    4948   4949   4950   4951
HID      033606     320    328    336    344    5173#
HIF      033632     316    324    332    340    5177#
HINUM    002436     446#   667    670    677    733    736    743    802    805    812    857    860    867
                    914    917    924    987    990    997    1064   1067   1074   1124   1127   1134   1185
                    1188   1195   1252   1255   1262   1321   1324   1331   1376   1379   1386   1433   1436
                    1443   1505   1508   1515   1581   1584   1591   1641   1644   1651   1702   1705   1712
                    1769   1772   1779   1838   1841   1848   1893   1896   1983   1949   1952   1959   2021
                    2024   2031   2097   2100   2107   2157   2160   2167   2218   2221   2228   2285   2288
                    2295   2354   2357   2364   2421   2424   2431   2490   2493   2500   2562   2565   2572
                    2638   2641   2648   2710   2713   2720   2786   2789   2796   2852   2855   2862   2921
                    2924   2931   2992   2995   3002   5173   5177
HOTNPM   030414     651#   653#   46438
HT     = 000011      388    4742   4783
IOIVEC=  000020     1338   529#   530#
LF     = 000012      398    498    503    511    4777   4783
LOD      033572     328    328    336    344    5171#
LOF      033622     316    324    332    340    5175#
LONUM    002426     444#   667    669    676    733    735    742    802    804    811    857    859    866
                    914    916    923    987    989    996    1064   1066   1073   1124   1126   1133   1185
                    1187   1194   1252   1254   1261   1321   1323   1330   1376   1378   1385   1433   1435
                    1442   1505   1507   1514   1581   1583   1590   1641   1643   1650   1702   1704   1711
                    1769   1771   1778   1838   1840   1847   1893   1895   1902   1949   1951   1958   2021
                    2023   2030   2097   2099   2106   2157   2159   2166   2218   2220   2227   2285   2287
                    2294   2354   2356   2363   2421   2423   2430   2490   2492   2499   2562   2564   2571
                    2638   2640   2647   2710   2712   2719   2786   2788   2795   2852   2854   2861   2921
                    2923   2938   2992   2994   3001   5171   5175
MED    = 076600     142#   580    632    636    646    3259   3267   3273   4980
MEND1    011682     1741   1748   1752#
MEND10   014156     2184   2193   2200   22041
MEND2    012076     1808   1817   1821#
MEND3    012324     1865   1872   1876#
MEND4    012562     1920   1929   1933#
MEND5    013070     1980   1995   2000   2004#
```



```
MEND6    013416     2060   2069   2076   2080#
MEND7    013666     2124   2131   2136   2140#
MERR1    011502     1723   1727#
MERR2    011766     1790   1794#
MERR5    012746     1970   1974#
MERR6    013254     2042   2046#
MRET1    011450     1694   1720#
MRET10   014026     2149   2175#
MRET2    011742     1761   1787#
MRET3    012236     1830   1856#
MRET4    012464     1885   1911#
MRET5    012722     1941   1967#
MRET6    013230     2013   2039#
MRET7    013556     2089   2115#
MTST1    011540     1728   1736   1739#
MTST10   014052     2178   2182#
MTST2    012024     1795   1803   1806#
MTST3    012262     1859   1863#
MTST4    012510     1914   1918#
MTST5    013004     1975   1983   1986#
MTST6    013312     2047   2055   2058#
MTST7    013602     2110   2122#
MULC0    002550      472#  3376   4004#
MULC1    002552      473#  3376   4035#
MULC2    002554      474#  3376   4091#  4122#
MULC3    002556      475#  3376   4094#  4125#
MULC4    002560      476#  3376   4000#
MULC5    002562      477#  3376   4083#
NEWPAS   003450      584    596    611#   3329
NWPAS1   002720      511#   618
OP1      002446     448#   3066   3075   3096   3154   3165   3186   5179   5191
OP1D     033722      421    5191#
OP1F     033642      417    5179#
OP2      002456     450#   3068   3076   3097   3156   3178   3189   5181   5193
OP2D     033736      421    5193#
OP2F     033652      417    5181#
OP3      002466     452#   3070   3078   3100   3156   3166   3187   5183   5195
OP3D     033752      421    5195#
OP3F     033662      417    5183#
OP4      002476     454#   3066   3079   3098   3154   3171   3190   5185   5197
OP4D     033766      421    5197#
OP4F     033672      417    5185#
OP5      002506     456#   3068   3084   3102   3158   3159   3168   3188   5187   5199
OP5D     034002      421    5199#
OP5F     033702      417    5187#
OP6      002516     458#   3070   3071   3080   3099   3158   3161   3173   3191   5189   5201
OP6D     034016      421    5201#
OP6F     033712      417    5189#
PIRQ   = 177772       45#
PIRQVE=  000240     139#
PR0    = 000000      62#
PR1    = 000040      63#
PR2    = 000100      64#
PR3    = 000140      65#
PR4    = 000200      66#
PR5    = 000240      67#
```

```
PR6     = 000300            68#
PR7     = 000340            69#
PS      = 177776            42#     43
PSW     = 177776            43#
PWRVEC= 000024             134#    535*    536*    4958*   4959*   4960*   4974*   4988*   4989*
RAND    023350            3500    3503#
RANDL2  023342             666     881     913     1063    1184    1320    1432    1580    1701    1837    1948    2096    2217
                          2353    2489    2637    2785    2920    3065    3067    3069    3501#
RANDL4  023332             732     856     906     1123    1251    1375    1504    1640    1768    1892    2020    2156    2284
                          2420    2561    2709    2851    2991    3153    3155    3157    3498#
RESVEC= 000010             129#
RFLAG = 000144            1464#    636     3267
RWHAMI= 000022             144#    580     632     3259
SCRLF   023314            3367    3305#
SHDP1   023317            3344    3306#
SHDP2   022664            3348    3427#
SHDP3   023055            3352    3448#
SHT     022312            3365    3384#
STACK = 001100              33#    527     611     630
START   000000             172     520#    4993
STATS   022122            3319    3340#
STKLMT= 177774             44#
SUBPAS  003516             614     616     630#    3277    3299
SVEC1   022230            3342    3372#
SVEC2   022254            3346    3376#
SVEC3   022272            3350    3379#
SWR     001144             237#    525     547*    549     555*    562*    613     638     641     3263    3313    3317    4508
                          4522    4530    4537    4583    4590    4602    4605    4966    4981*
SWREG   000176             170#    555
SW0     = 000001            97#
SW00    = 000001            87#     97      641
SW01    = 000002            96#     96      638     3263
SW02    = 000004            85#     95
SW03    = 000010            84#     94
SW04    = 000020            83#     93
SW05    = 000040            82#     92
SW06    = 000100            81#     91
SW07    = 000200            80#     90
SW08    = 000400            79#     89
SW09    = 001000            78#     88
SW1     = 000002            96#
SW10    = 002000            77#    3313
SW11    = 004000            76#
SW12    = 010000            75#    3317
SW13    = 020000            74#
SW14    = 040000            73#
SW15    = 100000            72#
SW2     = 000004            95#
SW3     = 000010            94#
SW4     = 000020            93#
SW5     = 000040            92#
SW6     = 000100            91#
SW7     = 000200            90#
SW8     = 000400            89#
SW9     = 001000            88#
TBITVE= 000014             130#
```



```
TKVEC   = 000060            137#
TPVEC   = 000064            138#
TRAPVE= 000034             136#    533*    534*
TRTVEC= 000014             131#
TST1    023622             658#
TST10   006172            1115#
TST11   006462            1176#
TST12   006746            1243#
TST13   007242            1312#
TST14   007470            1367#
TST15   007726            1424#
TST16   010234            1496#
TST17   010562            1572#
TST2    004106              724#
TST20   011032            1632#
TST21   011322            1693#
TST22   011606            1760#
TST23   012102            1829#
TST24   012330            1884#
TST25   012566            1940#
TST26   013074            2012#
TST27   013422            2088#
TST3    004402              793#
TST30   013672            2148#
TST31   014162            2209#
TST32   014446            2276#
TST33   014742            2345#
TST34   015226            2412#
TST35   015522            2481#
TST36   016034            2553#
TST37   016356            2629#
TST4    004630              848#
TST40   016664            2701#
TST41   017212            2777#
TST42   017476            2843#
TST43   017772            2912#
TST44   020300            2983#
TST45   020626            3055#
TST46   021204            3143#
TST5    005066              905#
TST6    005374              976#
TST7    005722            1055#
TYPE    = 104401             575     593     585     618     623     3315    3344    3348    3352    3365    3367    4585    4593
                          4642    4663    4665    4666    4669    4671    4674    4677    4684    4686    4691    4747    4900
                          4948#   4990
TYPOC   = 104402            4651    4673    4676    4681    4949#
TYPON   = 104404            4951#
TYPOS   = 104403             621    3360    4958*
WFLAG   = 000344             147#    646     3273
XHI     034042              5173    5177    5208#
XLO     034032              5171    5175    5206#
XOP1    034052              5179    5191    5210#
XOP2    034060              5181    5193    5211#
XOP3    034066              5183    5195    5212#
XOP4    034074              5185    5197    5213#
XOP5    034110              5187    5199    5215#
```

```
XOP6    #34102  5189    5201    5214#
$ADD    #23566   671     737     806     861     918     991    1060    1128    3082    3167    3671#
$APTHD  #01000   194     200#
$ASTAT= ****** U 4817    4832
$ATYC   #31160  4788    4790#
$ATY1   #31134  4786#
$ATYJ   #31142  4732    4787#
$ATY4   #31152  4598    4789#
$AUTOB  #01140   233#
$BDADP  #01126   228#
$BDDAT  #01132   230#
$BELL   #01172   250#    3315    4585    4617
$CHARC  #31130  4749#    4759#    4766    4775#    4780#
$CKSWR= ****** U 4954
$CMTAG  #01100   215#     522     523     531     537     538     539
$CM3    = 000000 248#
$CONV   #27622  3675    4008    4267    4483#
$CPUOP  #01230   273#
$CRLF   #01177   252#     623    4593    4617    4643    4665    4671    4686    4748    4783
$DEVCT  #01212   264#
$DIV    #26364  2222    2289    2358    2425    2494    2566    2642    2714    2790    2856    2925    2996    3081
                3169    3174    4263#
$DOAGN  #22116  3308    3322    3329#
$ENDAD  #22106   188    3324#    4612
$ENDCT  #22042   537     615    3310#
$ENV    #01222   269#    4595    4727    4795    4819
$ENVM   #01223   270#     560    4729    4734    4797
$EOP    #21764  3261    3264    3276    3291#
$EOPCT  #22034   537#     615    3307#    3311
$ERFLG  #01104   219#    3246#    3292#    4526    4528    4534#    4556    4580#    4617
$ERMAX  #01120   225#     540#    4520    4551#    4556
$ERROR  #30150   531     4570#
$ERPPC  #01122  2261#    4587#    4588#    4589    4617    4650    4698
$ERPTR  #01232   284#     4650
$ERTTL  #01114  2231#    4586#    4617
$ESCAP  #01170  2491     539#    4550#    4608    4610    4617
$ETABL  #01222   268#
$ETEND  #01232   286     2801
$FATAL  #01204  2611     4823#
$FEA    #02404   439#     662#     700     728#     766     797#     852#     909#     947     982#    1020    1059#    1119#
                1180#    1218    1247#    1285    1316#    1371#    1428#    1466    1500#    1538    1576#    1636#    1697#
                1735    1764#    1802    1833#    1888#    1944#    1982    2016#    2054    2092#    2152#    2213#    2251
                2280#    2318    2349#    2387    2416#    2454    2485#    2523    2557#    2595    2633#    2671    2705#
                2743    2781#    2819    2847#    2885    2910#    2954    2987#    3025    3060#    3148#    3827#    3847#
                3905#    4086#    4097#    4128#    4279#    4373#    4384#    4405#    5157    5165
$FEC    #02402   438#     661#     696     727#     762     796#     851#     908#     943     981#    1016    1058#    1118#
                1179#    1214    1246#    1281    1315#    1370#    1427#    1462    1499#    1534    1575#    1635#    1696#
                1731    1763#    1798    1832#    1887#    1943#    1978    2015#    2050    2091#    2151#    2212#    2247
                2279#    2314    2348#    2383    2415#    2450    2484#    2519    2556#    2591    2632#    2667    2704#
                2739    2780#    2815    2846#    2881    2915#    2950    2986#    3021    3059#    3147#    3826#    3846#
                3904#    4085#    4096#    4127#    4278#    4372#    4383#    4404#    5157    5165
$FFLG   #31400  4786#    4789#    4817    4826#    4834#
$FILIC  #01164   246#    4752    4783
$FILIS  #01163   245#    4783
$FPS    #02400   437#     660#     674     687     692     726#     740     753     758     795#     809     822     850#
                864     877     907#     921     934     939     982#     994    1007    1012    1057#    1071    1084
```



```
                1117#    1131    1144    1178#    1192    1205    1210    1245#    1259    1272    1277    1314#    1328
                1341    1369#    1383    1396    1426#    1440    1453    1458    1498#    1512    1525    1530    1574#
                1588    1601    1634#    1648    1661    1695#    1709    1722    1727    1762#    1776    1789    1794
                1831#    1845    1858    1886#    1900    1913    1942#    1956    1969    1974    2014#    2028    2041
                2046    2090#    2104    2117    2150#    2164    2177    2211#    2225    2238    2243    2278#    2292
                2305    2318    2347#    2361    2374    2379    2414#    2428    2441    2446    2483#    2497    2510
                2515    2555#    2569    2582    2587    2631#    2645    2658    2663    2703#    2717    2730    2735
                2779#    2793    2806    2811    2845#    2859    2872    2877    2914#    2928    2941    2946    2985#
                2999    3012    3017    3058#    3090    3105    3146#    3180    3195    3591    3605    3672#    3673
                3820#    3822    3825#    3840#    3842    3845#    3863#    3866#    3874    3899    3903#    4005#    4006
                4081    4084#    4090#    4092    4095#    4121#    4123    4126#    4141#    4144#    4152    4264#    4265
                4277#    4368    4371#    4377#    4379    4382#    4398#    4400    4403#    4415#    4418#    4426    5156
                5165
$GDADR  #01124   227#
$GDDAT  #01130   229#
$GET42  #22676  3318    3321#
$GTS**= ****** U 4953
$HD     = 000000  12
$HIBIS  #01000   201#
$ICNT   #01105   220#    4581#    4542    4544#    4555
$ILLUP  #32050  4958    4974    4995#
$INTAG  #01141   234#
$ITEMR  #01116   224#    4589#    4597    4617    4647
$LF     #01200   253#    4617    4783
$LFLG   #31377  4027#    4833#
$LPADR  #01110   221#     541#     680#     746#     815#     870#     927#    1000#    1077#    1137#    1198#    1265#    1334#
                1389#    1446#    1518#    1594#    1654#    1715#    1782#    1851#    1906#    1962#    2034#    2110#    2170#
                2231#    2298#    2367#    2434#    2503#    2575#    2651#    2723#    2799#    2865#    2934#    3005#    3091#
                3181#    4532#    4548#    4553    4555
$LPREG  #01112   222#     542#    4532    4549#    4555    4607
$LPTST  #01150   239#    4524
$MAIL   #01202   202     206     259#     559     4547    4595    4727
$MBADR  #01002   202#
$MFLG   #31376  4787#    4793    4828#    4832#
$MSGAD  #01216   266#    4803#    4806
$MSGLG  #01220   267#    4808#
$MSGTY  #01202   260#    4801    4809#    4821    4825#
$NUL    #25244  1706    1773    1842    1897    1953    2025    2101    2161    3083    3085    3175    4004#
$NXCNT  #30146  4545    4555#
$NULL   #01162   244#    4754    4783
$NWTST= #00003   655#     721#     790#     845#     902#     975#    1052#    1112#    1173#    1240#    1309#    1364#    1421#
                1493#    1569#    1629#    1690#    1757#    1826#    1881#    1937#    2009#    2085#    2145#    2206#    2273#
                2342#    2409#    2478#    2550#    2626#    2698#    2774#    2840#    2909#    2980#    3052#    3140#
$OCNT   #31624  4872#    4901#    4914#
$OMODE  #31626  4867#    4871#    4876    4879#    4890#    4916#
$OVAR   #30132  4509     4525    4533    4543    4552#
$PASS   #01210   263#     559#     619    3304#    3305#    4539    4556
$PASTM  #01006   204#
$POLSH  #23500   668     734     803     858     915     988    1065    1125    1186    1253    1322    1377    1434
                1506    1582    1642    1703    1770    1839    1894    1950    2022    2098    2158    2219    2286
                2355    2422    2491    2563    2639    2711    2787    2853    2922    2993    3074    3164    3585#
$POPX   #23540   672     738     807     862     919     992    1069    1129    1190    1257    1326    1381    1438
                1510    1596    1646    1707    1774    1843    1898    1954    2026    2102    2162    2223    2290
                2359    2426    2495    2567    2643    2715    2791    2857    2926    2997    3086    3176    3602#
$POWER  #32056  4991    4990#
$PUSH   #23510   669     670     735     736     804     805     859     860     916     917     989     990    1066
```

```
                    1067    1126    1127    1187    1198    1254    1255    1323    1324    1378    1379    1435    1436
                    1507    1508    1583    1584    1643    1644    1704    1705    1771    1772    1840    1841    1895
                    1896    1951    1952    2023    2024    2099    2100    2159    2160    2220    2221    2287    2288
                    2356    2357    2423    2424    2492    2493    2564    2565    2640    2641    2712    2713    2788
                    2789    2854    2855    2923    2924    2994    2995    3075    3076    3078    3079    3080    3084
                    3165    3166    3168    3170    3171    3173    3590#
$PWRAD  @32044      4993#
$PWRDN  @31670       535    4958#   4988
$PWRMG  @32040      4991#
$PWRUP  @31750      4968    4974#
$QUES   @01176       251#   4617    4783
$RDCHP= ****** U    4954
$RDDEC= ****** U    4954
$RDLIN= ****** U    4954
$RDOCT= ****** U    4954
$R2A  = ****** U    4954
$SAVRE= ****** U    4954
$SAVR6  @32054      4967#   4975    4976#   4977#   4997#
$SCOPE  @27672       529    4506#
$SETUP= @00037       514#    528     529     531     533     535     537     538     539     541    4507    4571    4605
                    4612
$STOP = 177777       514#
$SUB    @23562      1189    1256    1325    1388    1437    1509    1585    1645    3077    3172    3670#
$&VLAD  @30076      4517    4546#
$&VPC = @00204       178#    183
$&WR  = 167400         1#      12     248     249     250     538     539     541     542     659     725     794     849
                     906     979    1056    1116    1177    1244    1313    1368    1425    1497    1573    1633    1694
                    1761    1830    1885    1941    2013    2089    2149    2210    2277    2346    2413    2482    2554
                    2630    2702    2778    2844    2913    2984    3056    3144    4498    4499    4500    4501    4502
                    4508    4520    4522    4523    4526    4527    4528    4535    4536    4537    4549    4552    4555
                    4562    4563    4564    4565    4566    4583    4590    4602    4605    4617    4994
$&WREG  @01224       271#    562
$&WRMK= @00003      4502    4524
$TESTN  @01206       262#   4547#   4697
$TIMES  @01166       248#    538#   3245#   3294#   4535#   4542   4545#   4555
$TKB    @01154       241#
$TKS    @01152       240#
$TN   = @00047         1#      12     655     659#    721     725#    790     794#    845     849#    902     906#    975
                     979#   1052    1056#   1112    1116#   1173    1177#   1240    1244#   1309    1313#   1364    1368#
                    1421    1425#   1493    1497#   1569    1573#   1629    1633#   1690    1694#   1757    1761#   1826
                    1830#   1881    1885#   1937    1941#   2009    2013#   2085    2089#   2145    2149#   2206    2210#
                    2273    2277#   2342    2346#   2409    2413#   2478    2482#   2550    2554#   2626    2630#   2698
                    2702#   2774    2778#   2840    2844#   2909    2913#   2980    2984#   3052    3056#   3140    3144#
$TPB    @01160       243#   4772#   4783
$TPFLG  @01165       247#   4721    4783
$TPS    @01156       242#   4770    4783
$TRAP   @31630       533    4925#
$TRAP2  @31652      4936#   4947
$TRP  = @00005      4940#   4949#   4950#   4951#   4952#
$TRPAD  @31664      4930    4947#
$TSTM   @01006       203#
$TSTNM  @01102       218#   3247#   3293#   4524    4546#   4547    4552    4556    4582    4617
$TYPBN= ****** U    4952
$TYPDS= ****** U    4952
$TYPE   @30652      4721#   4814    4940    4948
$TYPEC  @31064      4751    4758    4765    4770#   4771
```



```
$TYPER  @30412      4592    4641#
$TYPEX  @31132      4776    4778    4781#
$TYPOC  @31126      4870#   4949
$TYPON  @31442      4869    4872#   4951
$TYPOS  @31402      4865#   4950
$UNIT   @01214       265#
$UNITM  @01010       205#
$USWR   @01226       272#
$XTSTP  @27702      4511#
$0FILL  @31625      4866#   4870#   4880    4915#
$40CAT= ****** U    4508    4592
.     = @34116       164#    168#    178     179#    181#    183#    184#    190     191#    193#    195#    214#    254
                     515#    526     541     542     680     746     815     870     927    1000    1077    1137    1198
                    1265    1334    1389    1446    1518    1594    1654    1715    1782    1851    1906    1962    2034
                    2110    2170    2231    2298    2367    2434    2503    2575    2651    2723    2799    2865    2934
                    3005    3091    3181    3471#   4555    4556    4617    4703#   4783    4835#   4970    4996    5155#
.$ASTA= ****** U    4787    4790
.$X   = @01000       190#    195
```

```
CKPFLT      1#
COMMEN    140#
COMM00      1#
COMM01      1#
COMM02      1#
COMM03      1#
COMM04      1#
COMM05      1#
COMM06      1#
COMM07      1#
COMM1       1#
COMM10      1#
COMM11      1#
COMM12      1#
COMM13      1#
COMM14      1#
COMM15      1#
COMM16      1#
COMM17      1#
COMM2       1#
COMM20      1#
COMM21      1#
COMM22      1#
COMM23      1#
COMM24      1#
COMM25      1#
COMM26      1#
COMM27      1#
COMM3       1#
COMM30      1#
COMM31      1#
COMM32      1#
COMM33      1#
COMM34      1#
COMM35      1#
COMM36      1#
COMM37      1#
COMM4       1#
COMM40      1#
COMM41      1#
COMM42      1#
COMM43      1#
COMM44      1#
COMM45      1#
COMM46      1#
COMM47      1#
ENDCOM    140#
ERRCMP      1#
ERRLOR      1#  4616
ERROR     34#   69#    690    702    715    756    764    768    783    825    838    880    895    937    945
         949    967   1010   1018   1022   1044   1087   1105   1147   1169   1208   1216   1220   1233   1275
        1283   1287   1302   1344   1357   1399   1414   1456   1464   1468   1486   1528   1536   1540   1562
        1604   1622   1664   1686   1725   1733   1737   1750   1792   1800   1804   1819   1861   1874   1916
        1931   1972   1980   1984   2002   2044   2052   2056   2078   2120   2138   2180   2202   2241   2249
        2253   2266   2308   2316   2320   2335   2377   2385   2389   2402   2444   2452   2456   2471   2513
        2521   2525   2543   2585   2593   2597   2619   2661   2669   2673   2691   2733   2741   2745   2767
```



```
        2809   2817   2821   2834   2875   2883   2887   2902   2944   2952   2956   2974   3015   3023   3027
        3049   3100   3133   3198   3238   3485
ESCAPE    140#
FCOM0       1#
FCOM1       1#
FCOM2       1#
FCOM3       1#
FCOM4       1#
FPRGT0      1#
FPRGT1      1#
FPSFEC      1#
FPSTST      1#
GENCOM      1#
GENTS1      1#
GENTS2      1#
GENTS3      1#
GENTS4      1#
GETPRI    140#
GETSWR    140#
GTSTD       1#
GTSTF       1#
HTSTD       1#
HTSTF       1#
MOVD1S      1#
MULT      140#
NEWTST    140#   655    721    790    845    902    975   1052   1112   1173   1240   1309   1364   1421   1493
         1569   1629   1690   1757   1826   1881   1937   2009   2085   2145   2206   2273   2342   2409   2478
         2550   2626   2698   2774   2840   2909   2980   3052   3140
POP       140#  4829   4830   4981   4982
PUSH      140#  4790   4792   4813   4960   4966
REPORT    140#
SBTST1      1#
SBTST2      1#
SCOM0       1#
SCOM1       1#
SCOM2       1#
SCOM3       1#
SCOM4       1#
SCOM5       1#
SCOM6       1#
SCOM7       1#
SCOPE     35#   658    724    793    848    905    978   1055   1115   1176   1243   1312   1367   1424   1496
         1572   1632   1693   1760   1829   1884   1940   2012   2088   2148   2209   2276   2345   2412   2481
         2553   2629   2701   2777   2843   2912   2983   3055   3143   3244
SCPLOR      1#  4587
SEADAT      1#
SETPRI    140#
SETREG      1#  4571
SETTRA  4940#  4949   4950   4951
SETUP       1#  1488   520
SKIP      140#
SLASH     140#   577    604
SPACE     140#
STARS     140#   176    187    189    196    210    254    257    516    518    607    609    626    628    655
         657    721    723    790    792    845    847    902    904    975    977   1052   1054   1112   1114
        1173   1175   1240   1242   1309   1311   1364   1366   1421   1423   1493   1495   1569   1571   1629
```

```
        1631    1690    1692    1757    1759    1826    1828    1881    1883    1937    1939    2009    2011    2085    2087
        2145    2147    2206    2208    2273    2275    2342    2344    2409    2411    2478    2480    2550    2552    2626
        2628    2698    2700    2774    2776    2848    2842    2909    2911    2980    2982    3052    3054    3093    3140
        3142    3193    3241    3280    4495    4558    4617    4706    4785    4842    4919    4956    4972
STATUS     1#
$WR5U    1408    5438
TADDD1     1#
TADDD2     1#
TADDF1     1#
TADDF2     1#
TADDR1     1#
TADDR2     1#
TRMTRP  4940#
TYPBIN   1408#
TYPDEC   1438#
TYPNAM   1408#
TYPNUM   1408#
TYPOCS   1408#
TYPOCT   1408#
TYPTXT   1481#
UPCODE     1#  4979
$$CMRE   2008#
$$CMTM   2081#
$$ESCA   1408#
$$NEWT   1408#   655     721     790     845     902     975    1052    1117    1173    1240    1309    1364    1421    1493
        1569    1629    1690    1757    1826    1881    1937    2009    2085    2145    2206    2273    2342    2409    2478
        2558    2626    2698    2774    2848    2909    2980    3052    3140
$$SET   4940#   4949    4950    4951
$$SETM   559#
$$SKIP   140#
.EQUAT     1#     30
.HEADE     1#      2
.SBPAS     1#   3241
.SETUP     1#    514
.STPAS     1#    577
.$ACT1     1#    174
.$APTB     1#   2558
.$APTH     1#    185
.$APTY     1#   4783
.$CATC     1#    162
.$CMTA     1#    208
.$EOP      1#
.$ERRO     1#   4556
.$POWE     1#   4954
.$SCOP     1#   4493
.$TRAP     1#   4917
.$TYER     1#
.$TYPE     1#   4784
.$TYPO     1#   4848

. ABS.  #34116     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0
```



```
 DQFPDB,BIN,DQFPDB,LST/CRF/SOL/P/DOC/CPU:70/EX/EN:WRP/NL:TTM:DQFPDB,MAC,DQFPDB.P11
 RUN-TIME:  19 16 .1 SECONDS
 RUN-TIME RATIO:  102/33=2.7
 CORE USED:  25K  (50 PAGES)

 DOCUMENT PAGES:  128
 WRAP-AROUND:  0%

 USER SYMBOLS:  686
 MACRO  NAMES:  141
 UNDF SYMBOLS:  14
 DISK BLOCKS  READ:  1230
 DISK BLKS WRITTEN:  629
 KILO CORE SECONDS:  1377
```