KD11-K

PDP11/6X FP11E FP RANDOM

MD-11-DQFPD-A

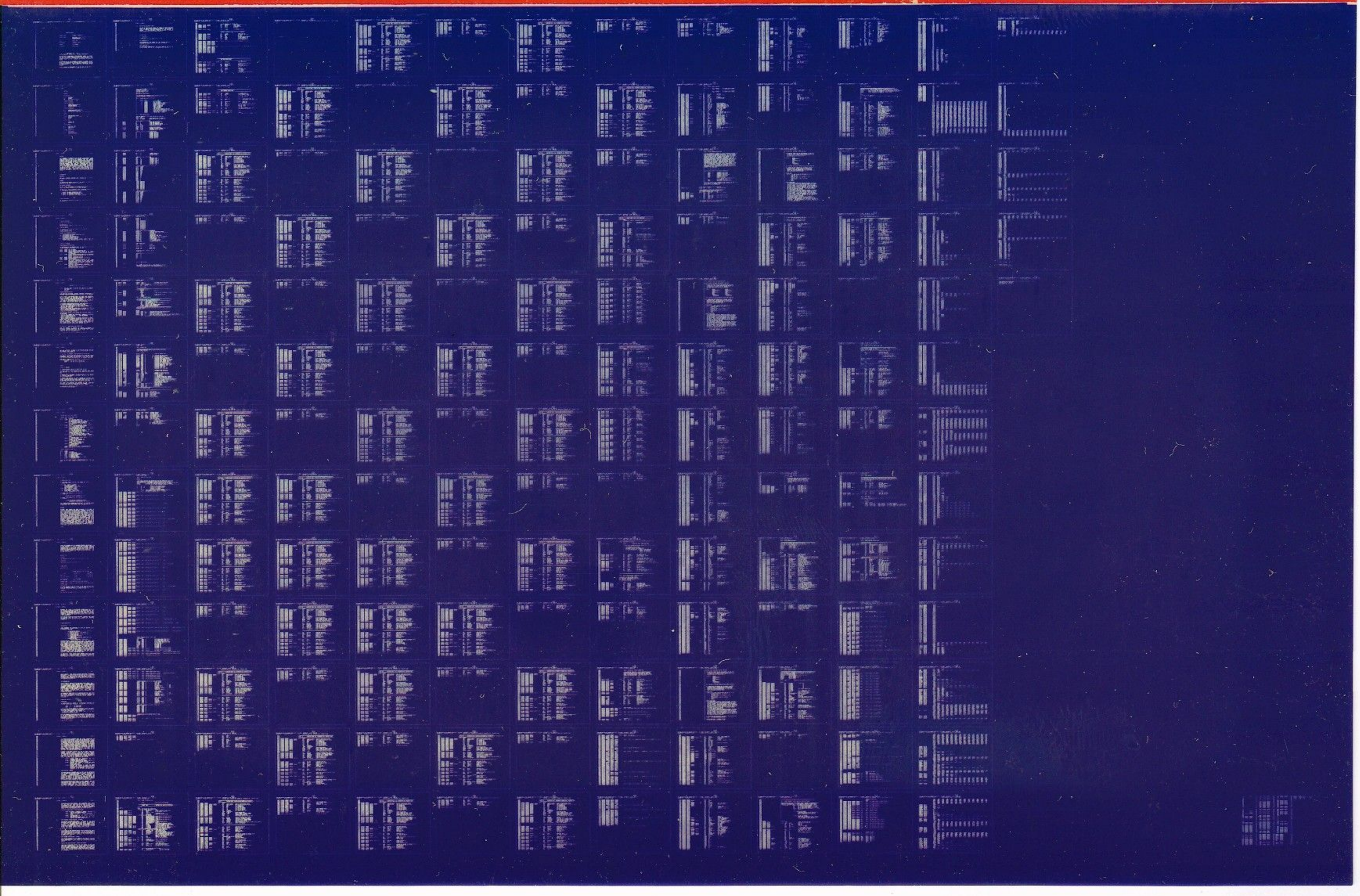
EP-DQFPD-A-DL-A

FICHE 1 OF 1

APR 1977

digital

MADE IN USA



```
                000000                  .REPT   0
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DQFPD-A-D
PRODUCT NAME:	PDP-11/6X - FP11-E FLOATING POINT UNIT ADD/SUB/MUL/DIV RANDOM OPERAND EXERCISER
DATE:	MARCH 1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	KEN CHAPMAN
REVISED BY:	DONALD 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 0-34164(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 "$LPTST" IF SET, THEN THE TEST SPECIFIED BY THE TEST NUMBER CONTAINED IN THE MEMORY WORD "$LPTST" (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 "$LPTST" MATCHES THE NUMBER OF THE TEST CURRENTLY EXECUTING. AT THIS POINT, THE TEST IS LOOPED ON INDEFINATELY, UNTIL EITHER SW<8>=0 OR "$LPTST" IS CHANGED. NOTE THAT IF "$LPTST" DOES NOT MATCH THE TEST NUMBER OF ANY TEST, THE CONTENTS OF "$LPTST" 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

AVERAGE EXECUTION TIME PER PASS

MODEL	SHORTEST PASS	LONGEST PASS
PDP-11/6X	20 SEC	8 MIN:30 SEC
PDP-11/6X W/FP11-E	XX SEC	X MIN:XX SEC

SEC = SECONDS / MIN = MINUTES

SHORTEST PASS ::= NO ITERATIONS, USING SWR=(004000)

LONGEST PASS ::= 2000(10) ITERATIONS/TEST, USING SWR=(000000)

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 CODF 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:

- EREG0 THRU EREG7 - 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 POUTINE 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 ($PWRDN) SAVES THE GENERAL REGISTERS AND STACK POINTER. THE POWER UP ROUTINE ($PWRUP) 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 - $EOP

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.

.ENDR

```
                                .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    000CC0          0=BELL ON PASS END
                                ;*              002000          1=BELL ON ERROR
                                ;*         9    001000          LOOP ON ERROR
                                ;*         8    000400          LOOP ON TEST NUMBER IN "SLPTST"
                                ;*         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
                                ;*              000003          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
```

```
                        R7=     %7              ;;GENERAL REGISTER
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        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)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        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          RWHAMI= 022             ;READ WHAMI

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 ".+2,HALT"
                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                  .=174
```

```
                                        TRAP CATCHER

000174  000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  002734                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000204                          $SVPC=.                 ;SAVE PC
        000046                          .=46
000046  022160                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
        000204                          .=$SVPC                 ;; RESTORE PC
        001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
        001000                          .$X=.   ;;SAVE CURRENT LOCATION
        000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
        001000                          .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;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  000776                  $TSTM:  .WORD   510.    ;;RUN TIM OF LONGEST TEST
001006  000001                  $PASTM: .WORD   1       ;;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
001100  000000                  .WORD   0
001102  000000          $TSTNM: .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          $LPTST: .WORD   0               ;;CONTAINS TEST NUMBER TO LOOP UPON
001114  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001116  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001120  000000          $ITEMB: .WORD   0               ;;CONTAINS ITEM CONTROL BYTE
001122  000001          $ERMAX: .WORD   1               ;;CONTAINS MAX. ERRORS PER TEST
001124  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001126  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001130  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001132  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001134  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001136  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001140  000000                  .WORD   0
001142  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001143  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001144  000000                  .WORD   0
001146  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001150  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
                        $TKS:   177560                  ;;TTY KBD STATUS
                        $TKB:   177562                  ;;TTY KBD BUFFER
                        $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
                        $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  000000          $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
```

APT MAILBOX-ETABLE

```
  1001  001216  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
  1002  001220  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
  1003  001222                        $ETABLE:                ;;APT ENVIRONMENT TABLE
  1004  001222     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
  1005  001223     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
  1006  001224  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
  1007  001226  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
  1008  001230  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
  1009                                ;*                      BITS 15-11=CPU TYPE
  1010                                ;*                             11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
  1011                                ;*                             11/70=06,PDQ=07,Q=10
  1012                                ;*                      BIT 10=REAL TIME CLOCK
  1013                                ;*                      BIT  9=FLOATING POINT PROCESSOR
  1014                                ;*                      BIT  8=MEMORY MANAGEMENT
  1015  001232                        $ETEND:
  1016                                .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 $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                        ;*      DH              ;;POINTS TO THE DATA HEADER
                                        ;*      DT              ;;POINTS TO THE DATA
                                        ;*      DF              ;;POINTS TO THE DATA FORMAT
                                                ;*NOTE:  ERROR VECTOR TABLE ($ERRTB) HAS BEEN MODIFIED,
                                                ;*       SEE $ERRTYP ROUTINE FOR ITS STRUCTURE
001232  032574  033254  033544  EMV001: .WORD   EMJ,DHA,DTA,0,0,0,0,0,0,0       ;ADDF    *
001240  000000  000000  000000
001246  000000  000000  000000
001254  000000
001256  032615  033254  033544  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  032636  033254  033544  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  032345  033270  033564  EMV005: .WORD   EME,DHB,DTD,LOF,HIF,0,0,0,0,0   ;ADDF    *
001360  033672  033702  000000
001366  000000  000000  000000
001374  000000
001376  032345  033326  033576  EMV006: .WORD   EME,DHC,DTE,LOD,HID,0,0,0,0,0   ;ADDD    *
001404  033642  033656  000000
001412  000000  000000  000000
001420  000000
001422  032375  033270  033564  EMV007: .WORD   EMF,DHB,DTD,LOF,HIF,0,0,0,0,0   ;SUBF    *
001430  033672  033702  000000
001436  000000  000000  000000
001444  000000
001446  032375  033326  033576  EMV010: .WORD   EMF,DHC,DTE,LOD,HID,0,0,0,0,0   ;SUBD    * RESULT ERRORS
001454  033642  033656  000000
001462  000000  000000  000000
001470  000000
001472  032425  033270  033564  EMV011: .WORD   EMG,DHB,DTD,LOF,HIF,0,0,0,0,0   ;MULF    *
001500  033672  033702  000000
001506  000000  000000  000000
001514  000000
001516  032425  033326  033576  EMV012: .WORD   EMG,DHC,DTE,LOD,HID,0,0,0,0,0   ;MULD    *
001524  033642  033656  000000
```

```
1073  001532  000000  000000  000000
1074  001540  000000
1075  001542  032455  033270  033564  EMV013: .WORD   EMH,DHB,DTD,LOF,HIF,0,0,0,0,0   ;DIVF   *
1076  001550  033672  033702  000000
1077  001556  000000  000000  000000
1078  001564  000000
1079  001566  032455  033326  033576  EMV014: .WORD   EMH,DHC,DTE,LOD,HID,0,0,0,0,0   ;DIVD   *
1080  001574  033642  033656  000000
1081  001602  000000  000000  000000
1082  001610  000000
1083
1084  001612  032505  033424  033620  EMV015: .WORD   EMI,DHD,DTF,0,0,0,0,0,0,0       ;ILLEGAL FPP TRAP
1085  001620  000000  000000  000000
1086  001626  000000  000000  000000
1087  001634  000000
1088
1089  001636  032657  033254  033544  EMV016: .WORD   EMM,DHA,DTA,0,0,0,0,0,0,0       ;DIVF   *
1090  001644  000000  000000  000000
1091  001652  000000  000000  000000
1092  001660  000000
1093  001662  032700  033254  033544  EMV017: .WORD   EMN,DHA,DTA,0,0,0,0,0,0,0       ;ADDD   *
1094  001670  000000  000000  000000
1095  001676  000000  000000  000000
1096  001704  000000
1097  001706  032721  033254  033544  EMV020: .WORD   EMO,DHA,DTA,0,0,0,0,0,0,0       ;SUBD   * FPS ERRORS
1098  001714  000000  000000  000000
1099  001722  000000  000000  000000
1100  001730  000000
1101  001732  032742  033254  033544  EMV021: .WORD   EMP,DHA,DTA,0,0,0,0,0,0,0       ;MULD   *
1102  001740  000000  000000  000000
1103  001746  000000  000000  000000
1104  001754  000000
1105  001756  032763  033254  033544  EMV022: .WORD   EMQ,DHA,DTA,0,0,0,0,0,0.0       ;DIVD   *
1106  001764  000000  000000  000000
1107  001772  000000  000000  000000
1108  002000  000000
1109
1110  002002  033004  033503  033552  E.   .3: .WORD   EMR,DHE,DTB,0,0,0,0,0,0,0       ;ADDF   *
1111  002010  000000  000000  000000
1112  002016  000000  000000  000000
1113  002024  000000
1114  002026  033031  033503  033552  EMV024: .WORD   EMS,DHE,DTB,0,0,0,0,0,0,0       ;SUBF   *
1115  002034  000000  000000  000000
1116  002042  000000  000000  000000
1117  002050  000000
1118  002052  033056  033503  033552  EMV025: .WORD   EMT,DHE,DTB,0,0,0,0,0,0,0       ;MULF   *
1119  002060  000000  000000  000000
1120  002066  000000  000000  000000
1121  002074  000000
1122  002076  033103  033503  033552  EMV026: .WORD   EMU,DHE,DTB,0,0,0,0,0,0,0       ;DIVF   * FEC/FEA ERRORS
1123  002104  000000  000000  000000
1124  002112  000000  000000  000000
1125  002120  000000
1126  002122  033130  033503  033552  EMV027: .WORD   EMV,DHE,DTB,0,0,0,0,0,0,0       ;ADDD   *
1127  002130  000000  000000  000000
1128  002136  000000  000000  000000
```

```
1129  002144  000000
1130  002146  033155  033503  033552  EMV030: .WORD   EMW,DHE,DTB,0,0,0,0,0,0,0        ;SUBD   *
1131  002154  000000  000000  000000
1132  002162  000000  000000  000000
1133  002170  000000
1134  002172  033202  033503  033552  EMV031: .WORD   EMX,DHE,DTB,0,0,0,0,0,0,0        ;MULD   *
1135  002200  000000  000000  000000
1136  002206  000000  000000  000000
1137  002214  000000
1138  002216  033227  033503  033552  EMV032: .WORD   EMY,DHE,DTB,0,0,0,0,0,0,0        ;DIVD   *
1139  002224  000000  000000  000000
1140  002232  000000  000000  000000
1141  002240  000000
1142
1143  002242  032153  033254  033544  EMV033: .WORD   EMA,DHA,DTA,0,0,0,0,0,0,0        ;F-MODE EXERCISER  *  FPS
1144  002250  000000  000000  000000
1145  002256  000000  000000  000000
1146  002264  000000
1147  002266  032210  033254  033544  EMV034: .WORD   EMB,DHA,DTA,0,0,0,0,0,0,0        ;D-MODE EXERCISER  *    ERROR
1148  002274  000000  000000  000000
1149  002302  000000  000000  000000
1150  002310  000000
1151
1152  002312  032245  033270  033564  EMV035: .WORD   EMC,DHB,DTD,OP1F,OP2F,OP3F,OP4F,OP5F,OP6F,0     ;F-MODE EXERCISER  *  RE
1153  002320  033712  033722  033732
1154  002326  033742  033752  033762
1155  002334  000000
1156  002336  032305  033326  033576  EMV036: .WORD   EMD,DHC,DTE,OP1D,OP2D,OP3D,OP4D,OP5D,OP6D,0     ;D-MODE EXERCISER  *
1157  002344  033772  034006  034022
1158  002352  034036  034052  034066
1159  002360  000000
1160
1161
1162
1163                                  .SBTTL  PROGRAM DEFINED COMMON TAGS
1164                                  ;*VARIABLES
1165  002362  000000                  FPS:    .WORD   0               ;FPS STORED HERE AFTER STFPS
1166  002364  000000                  FEC:    .WORD   0               ;FEC STORED HERE AFTER STST
1167  002366  000000                  FEA:    .WORD   0               ;FEA STORED HERE AFTER STST
1168  002370  000000                  FPPOPC: .WORD   0               ;OLD PC SAVED HERE AFTER TRAP
1169  002372  000000                  FPPOPS: .WORD   0               ;OLD PS SAVED HERE AFTER TRAP
1170  002374  000000                  FPPOSP: .WORD   0               ;SP AFTER TRAP
1171  002376  000000                  EXPFEA: .WORD   0               ;EXPECTED FEA
1172  002400  000000                  $FPS:   .WORD   0               ;SOFTWARE FPS
1173  002402  000000                  $FEC:   .WORD   0               ;SOFTWARE FEC
1174  002404  000000                  $FEA:   .WORD   0               ;SOFTWARE FEA
1175  002406  000000  000000  000000  ANS1:   .WORD   0,0,0,0         ;HARDWARE FLOATING POINT ANSWER
1176  002414  000000
1177  002416  000000  000000  000000  ANS2:   .WORD   0,0,0,0         ;SOFTWARE FLOATING POINT ANSWER
1178  002424  000000
1179  002426  027005  104552  111730  LONUM:  .WORD   027005,104552,111730,000555    ;RANDOM EXERCISER OPERAND LONUM
1180  002434  000555
1181  002436  102337  166330  007025  HINUM:  .WORD   102337,166330,007025,021553    ;RANDOM EXERCISER OPERAND HINUM
1182  002444  021553
1183  002446  175463  030712  105726  OP1:    .WORD   175463,030712,105726,124064    ;
1184  002454  124064
```

```
1185  002456  156607  002361  070707  OP2:    .WORD   156607,002361,070707,061111     ;
1186  002464  061111
1187  002466  003505  134261  062451  OP3:    .WORD   003505,134261,062451,052525     ;EXTRA RANDOM OPERANDS
1188  002474  052525
1189  002476  102466  123456  111111  OP4:    .WORD   102466,123456,111111,101010     ;
1190  002504  101010
1191  002506  076103  000346  060612  OP5:    .WORD   076103,000346,060612,125252     ;
1192  002514  125252
1193  002516  044100  112400  177777  OP6:    .WORD   044100,112400,177777,000006     ;
1194  002524  000006
1195
1196                                  ;*ADDED COUNTERS FOR TRACING WHERE WE'VE BEEN
1197  002526  000000                  ADDC0:  .WORD   0               ;TOTAL NUMBER           *
1198  002530  000000                  ADDC1:  .WORD   0               ;A=0, B#0               *
1199  002532  000000                  ADDC2:  .WORD   0               ;A#0, B=0               *
1200  002534  000000                  ADDC3:  .WORD   0               ;A=0, B=0               *  FOR
1201  002536  000000                  ADDC4:  .WORD   0               ;NO SHIFT REQ'D         *  ADD
1202  002540  000000                  ADDC5:  .WORD   0               ;OVERFLOW               *  AND
1203  002542  000000                  ADDC6:  .WORD   0               ;OVERFLOW, ENABLED      *  SUBTRACT
1204  002544  000000                  ADDC7:  .WORD   0               ;UNDERFLOW              *
1205  002546  000000                  ADDC8:  .WORD   0               ;UNDERFLOW, ENABLED     *
1206
1207  002550  000000                  MULC0:  .WORD   0               ;TOTAL NUMBER           *
1208  002552  000000                  MULC1:  .WORD   0               ;A=0 AND/OR B=0         *
1209  002554  000000                  MULC2:  .WORD   0               ;OVERFLOW               *  FOR
1210  002556  000000                  MULC3:  .WORD   0               ;OVERFLOW, ENABLED      *  MULTIPLY
1211  002560  000000                  MULC4:  .WORD   0               ;UNDERFLOW              *
1212  002562  000000                  MULC5:  .WORD   0               ;UNDERFLOW, ENABLED     *
1213
1214  002564  000000                  DIVC0:  .WORD   0               ;TOTAL NUMBER           *
1215  002566  000000                  DIVC1:  .WORD   0               ;NUM A=0                *
1216  002570  000000                  DIVC2:  .WORD   0               ;DEN B=0                *  FOR
1217  002572  000000                  DIVC3:  .WORD   0               ;OVERFLOW               *  DIVIDE
1218  002574  000000                  DIVC4:  .WORD   0               ;OVERFLOW, ENABLED      *
1219  002576  000000                  DIVC5:  .WORD   0               ;UNDERFLOW              *
1220  002600  000000                  DIVC6:  .WORD   0               ;UNDERFLOW, ENABLED     *
1221
1222                                  ;*REGISTER CONTENTS, AT ERROR, STORED HERE
1223  002602  000000                  EREG0:  .WORD   0
1224  002604  000000                  EREG1:  .WORD   0
1225  002606  000000                  EREG2:  .WORD   0
1226  002610  000000                  EREG3:  .WORD   0
1227  002612  000000                  EREG4:  .WORD   0
1228  002614  000000                  EREG5:  .WORD   0
1229  002616  000000                  EREG6:  .WORD   0
1230  002620  000000                  EREG7:  .WORD   0
1231
1232                                  ;*MESSAGES FOR BEGIN PROGRAM/START OF PASS
1233  002622  005015  005012  042115  BGNMES: .ASCII  <15><12><12><12>"MD-11-DQFPD-A..."
1234  002630  030455  026461  050504
1235  002636  050106  026504  027101
1236  002644  027056
1237  002646  042120  026520  030461          .ASCIZ  "PDP-11/6X F.P.U. ADD/SUB/MUL/DIV EXERCISER"<15><12>
1238  002654  033057  020130  027106
1239  002662  027120  027125  040440
1240  002670  042104  051457  041125
```

```
  1241  002676  046457  046125  042057
  1242  002704  053111  042440  042530
  1243  002712  041522  051511  051105
  1244  002720  005015     000
  1245  002723     015  050012  051501  NWPAS1: .ASCIZ  <15><12>"PASS #"
  1246  002730  020123  000043
```

```
                                       .SBTTL  START OF PASS ROUTINE

                                               .EVEN                   ;START ON AN EVEN BOUNDARY

                                       ;;**************************************************************************
                                               .ENABL  AMA             ; ASSEMBLE ALL RELATIVE REFERENCES AS ABSOLUTE
                                       ;;**************************************************************************

002734                                 START:
                                       .SBTTL  INITIALIZE THE COMMON TAGS
                                       ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002734  012706  001100                         MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002740  005026                               1: CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002742  022706  001146                         CMP     #SWR,R6 ;;DONE?
002746  001374                               2 BNE     .-6             ;;LOOP BACK IF NO
002750  012706  001100                         MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                       ;;INITIALIZE A FEW VECTORS
002754  012737  027742  000020               MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002762  012737  000340  000022               MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002770  012737  030220  000030               MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002776  012737  000340  000032               MOV     #340,@#EMTVEC+2 ;;LEVEL 7
003004  012737  031700  000034               MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
003012  012737  000340  000036               MOV     #340,@#TRAPVEC+2;LEVEL 7
003020  012737  031746  000024               MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
003026  012737  000340  000026               MOV     #340,@#PWRVEC+2 ;;LEVEL 7
003034  013737  022114  022106               MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
003042  005037  001166                       CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
003046  005037  001170                       CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
003052  012737  000001  001122               MOV     #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
003060  012737  003060  001110               MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
003066  012737  003066  001114               MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                       ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
                                       ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
003074  013746  000004                       MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
003100  012737  003134  000004               MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
003106  012737  177570  001146               MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
003114  012737  177570  001150               MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
003122  022777  177777  176016               CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
003130  001012                               BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                       ;;AND  THE HARDWARE SWR IS NOT = -1
003132  000403                               BR      65$             ;;BRANCH IF NO TIMEOUT
003134  012716  003142          64$:        MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
003140  000002                               RTI
003142  012737  000176  001146  65$:        MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
003150  012737  000174  001150               MOV     #DISPREG,DISPLAY
003156  012637  000004          66$:        MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

003162  005037  001210                       CLR     $PASS           ;;CLEAR PASS COUNT
003166  132737  000200  001223               BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
003174  001403                               BEQ     67$             ;;YES,USE NON-APT SWITCH
003176  012737  001224  001146               MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
003204                                 67$:

                                       ;SET UP FPP UNEXPECTED TRAP CATCHER - - - - - - - - -
003204  012737  023330  000244               MOV     #FPPILT,@#FPPVEC        ;NEW PC AT FPP TRAP
003212  005037  000246                       CLR     @#FPPVEC+2              ;NEW PS AT FPP TRAP
```

```
                                        INITIALIZE THE COMMON TAGS

1303
1304                                            ;*CLEAR COUNTERS FOR TRACING INFO
1305  003216  012700  002526                    MOV     #ADDCO,R0       ;FIRST LOCATION TO CLEAR
1306  003222  012701  000026                    MOV     #26,R1          ;26(8) WORDS
1307  003226  005020                  2$:      CLR     (R0)+           ;CLEAR AND BUMP
1308  003230  077102                           SOB     R1,2$           ;DO AGAIN
1309
1310  003232  104401  002622                    TYPE    ,BGNMES         ;ID MESSAGE AT START
1311
1312                                            ;/// ////////////////////////////////////////////////////////////////
1313                                            ;       MESSAGE ON WHETHER OR NOT HFP UNIT IS PRESENT
1314
1315  003236  076600  000022                    MFD     ,RWHAMI                         ;WHAMI INTO R0
1316  003242  032700  000020                    BIT     #BIT04,R0                       ;IS THERE A HFP UNIT ?
1317  003246  001403                           BEQ     70$                             ;NO, BR
1318  003250  104401  003264                    TYPE    ,68$                            ;INDICATE FP11-E PRESENT
1319  003254  000453                           BR      NEWPAS                          ;GO FOR SUBPASS INIT
1320  003256  104401  003324          70$:     TYPE    ,69$                            ;INDICATE NO FP11-E
1321  003262  000450                           BR      NEWPAS                          ;GO FOR SUBPASS INIT
1322
1323  003264  005015  020052  050106  68$:     .ASCIZ  <15><12>"* FP11-E HFP UNIT PRESENT *"<15><12>
1324  003272  030461  042455  044040
1325  003300  050106  052440  044516
1326  003306  020124  051120  051505
1327  003314  047105  020124  006452
1328  003322  000012
1329  003324  005015  020052  047516  69$:     .ASCIZ  <15><12>"* NO FP11-E HFP UNIT - ALL TESTS WFP ONLY *"<15><12>
1330  003332  043040  030520  026461
1331  003340  020105  043110  020120
1332  003346  047125  052111  026440
1333  003354  040440  046114  052040
1334  003362  051505  051524  053440
1335  003370  050106  047440  046116
1336  003376  020131  006452  000012
1337                                            .EVEN
1338
1339                                            ;////////////////////////////////////////////////////////////////
1340
1341
1342                                            ;;**************************************************************
1343                                            ;       NEW PASS ENTERS HERE
1344                                            ;;**************************************************************
1345
1346  003404  012706  001100          NEWPAS: MOV     #STACK,SP                       ;RESET STACK PTR
1347
1348  003410  032777  010000  175530            BIT     #BIT12,@SWR                     ;INHIBIT STATUS TYPEOUTS ?
1349  003416  001015                           BNE     SUBPAS                          ;BR IF YES
1350  003420  023737  022114  022106            CMP     $ENDCT,$EOPCT                   ;TIME FOR A MESSAGE ?
1351  003426  001011                           BNE     SUBPAS                          ;NO, NOT YET
1352
1353  003430  104401  002723                    TYPE    ,NWPAS1                         ;"PASS #"
1354  003434  013746  001210                    MOV     $PASS,-(SP)                     ;PASS COUNT INTO ...
1355  003440  005216                           INC     (SP)                            ;  1-N RANGE
1356  003442  104403                           TYPOS                                   ;TYPE OCTAL
1357  003444     006     000                   .BYTE   6,0                             ;  6 DIGITS, NO LEADING ZEROS
1358  003446  104401  001177                    TYPE    ,$CRLF                          ;END THE LINE
```

```
1359
1360
1361                                    ;;*****************************************************************
1362                                    ;NEW SUBPASS ENTERS HERE
1363                                    ;;*****************************************************************
1364
1365  003452  076600  000022           SUBPAS: MFD     ,RWHAMI                 ;GET WHAMI INTO R0
1366  003456  032700  000020                   BIT     #BIT04,R0               ;1=HFP PRESENT, 0=NO
1367  003462  001430                           BEQ     20$                     ;IF NO HFP, TEST WARM ONLY
1368
1369  003464  076600  000144                   MFD     ,RFLAG                  ;GET FLAGS INTO R0
1370
1371  003470  032777  000002  175450           BIT     #SW01,@SWR              ;SW01:  1=HFP OR WFP TEST ONLY
1372  003476  001413                           BEQ     1$                      ;       0=ALTERNATE HFP/WFP PER PASS
1373
1374  003500  032777  000001  175440           BIT     #SW00,@SWR              ;SW00:  1=WFP ONLY
1375  003506  001403                           BEQ     2$                      ;       0=HFP ONLY
1376  003510  042700  010000                   BIC     #BIT12,R0               ;CLEAR HFP ENABLE FLAG<5> FOR WFP
1377  003514  000402                           BR      3$                      ;
1378  003516  052700  010000           2$:     BIS     #BIT12,R0               ;SET HFP ENABLE FLAG<5> FOR HFP
1379  003522  076600  000344           3$:     MFD     ,WFLAG                  ;REWRITE FLAGS
1380
1381  003526  032700  010000           1$:     BIT     #BIT12,R0               ;TEST WHO'S ENABLED: HOT, WARM
1382  003532  001404                           BEQ     20$                     ;SET APPROPRIATE HEADER:
1383
1384  003534  012737  032136  030464   19$:    MOV     #ASCHOT,HOTWRM          ;"HOT: "
1385  003542  000403                           BR      21$                     ;
1386  003544  012737  032144  030464   20$:    MOV     #ASCWRM,HOTWRM          ;"WARM: "
1387  003552  005037  001102           21$:    CLR     $TSTNM                  ;ALL DONE, RESET TEST NUMBER COUNTER
```

```
                                                T1      EXERCISE ADDF, ALL INTERRUPTS ON, ROUNDING MODE

1388                                            ;;*****************************************************************
1389                                            ;*TEST 1         EXERCISE ADDF, ALL INTERRUPTS ON, ROUNDING MODE
1390                                            ;*****************************************************************
1391  003556  000004                           TST1:   SCOPE
1392  003560  012737  003712  002376                   MOV     #ARET1,EXPFEA   ;ADDR OF INSTR BEING TESTED
1393  003566  012737  007400  002400                   MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1394  003574  005037  002402                          CLR     $FEC            ;CLR FORTRAN FEC
1395  003600  005037  002404                          CLR     $FEA            ;CLR FORTRAN FEA
1396  003604  005037  002362                          CLR     FPS             ;CLR FPU FPS BUFFER
1397  003610  005037  002364                          CLR     FEC             ;CLR FPU FEC BUFFER
1398  003614  005037  002366                          CLR     FEA             ;CLR FPU FEA BUFFER
1399  003620  004737  023414                          JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1400  003624  002426  002436                          .WORD   LONUM,HINUM     ;
1401  003630  004437  023560                          JSR     R4,$POLSH       ;ENTER POLISH MODE
1402  003634  023562  002426                          $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1403  003640  023562  002436                          $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1404  003644  023640                                  $ADD                    ;ADDRESS OF FORTRAN ADD
1405  003646  023612  002416                          $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1406
1407  003652  013700  002400                          MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1408  003656  170127  040000                          LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
1409  003662  172437  002426                          LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1410  003666  172537  002436                          LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1411  003672  172737  002416                          LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1412  003676  170127  007400                          LDFPS   #007400         ;TURN INTERRUPTS ON
1413  003702  012737  003710  001110                   MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1414
1415                                            ;*****************************************************************
1416
1417  003710  172600                                  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1418  003712  172201                           ARET1:  ADDF    AC1,AC2         ;ADD AC1 BY AC2
1419  003714  170237  002362                          STFPS   FPS             ;STORE FLOATING POINT STATUS
1420  003720  023737  002362  002400                   CMP     FPS,$FPS        ;CHECK FPS
1421  003726  001403                                  BEQ     AERR1           ;BRANCH IF OK
1422  003730  174237  002406                          STF     AC2,ANS1        ;SAVE FPU ANSWER
1423  003734  104001                                  ERROR   1               ;FPS ERROR
1424
1425  003736  005737  002400                   AERR1: TST     $FPS            ;ERROR BIT SET ?
1426  003742  100014                                  BPL     ATST1           ;NO, DONT GET FEC/FEA
1427  003744  170337  002364                          STST    FEC             ;YES, CHECK STATUS
1428
1429  003750  023737  002364  002402                   CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1430  003756  001401                                  BEQ     1$              ;BRANCH IF OK
1431  003760  104023                                  ERROR   23              ;FEC IS WRONG
1432
1433  003762  023737  002366  002404   1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
1434  003770  001401                                  BEQ     ATST1           ;BRANCH IF OK
1435  003772  104023                                  ERROR   23              ;WRONG ADDRESS IN FEA
1436
1437  003774  173702                           ATST1: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1438  003776  170000                                  CFCC                    ;COPY FLOATING CONDITION CODES
1439  004000  001416                                  BEQ     AEND1           ;ANSWERS CHECK
1440                                                  ;COMPENSATE FOR FORTRAN INACCURACIES.
1441  004002  174237  002406                          STF     AC2,ANS1        ;SAVE FPU ANSWER
1442  004006  162737  000001  002410                   SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1443  004014  005637  002406                          SBC     ANS1
```

```
  1444  004020  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
  1445  004024  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
  1446  004026  001403                        BEQ     AEND1           ;BRANCH IF OK
  1447  004030  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
  1448  004034  104005                        ERROR   5               ;FPU AND FORTRAN DISAGREE
  1449
  1450  004036  005037  002362        AEND1:  CLR     FPS             ;CLR FPU FPS BUFFER
  1451
```

```
1452                                        ;;*****************************************************************
1453                                        ;*TEST 2        EXERCISE ADDD, ALL INTERRUPTS ON, ROUNDING MODE
1454                                        ;;*****************************************************************
1455  004042  000004                       TST2:   SCOPE
1456  004044  012737  004176  002376               MOV     #ARET2,EXPFEA   ;ADDR OF INSTR BEING TESTED
1457  004052  012737  007600  002400               MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1458  004060  005037  002402                     CLR     $FEC            ;CLR FORTRAN FEC
1459  004064  005037  002404                     CLR     $FEA            ;CLR FORTRAN FEA
1460  004070  005037  002362                     CLR     FPS             ;CLR FPU FPS BUFFER
1461  004074  005037  002364                     CLR     FEC             ;CLR FPU FEC BUFFER
1462  004100  005037  002366                     CLR     FEA             ;CLR FPU FEA BUFFER
1463  004104  004737  023404                     JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
1464  004110  002426  002436                     .WORD   LONUM,HINUM     ;
1465  004114  004437  023560                     JSR     R4,$POLSH       ;ENTER POLISH MODE
1466  004120  023562  002426                     $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
1467  004124  023562  002436                     $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
1468  004130  023640                           $ADD                    ;ADDRESS OF FORTRAN ADD
1469  004132  023612  002416                     $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
1470
1471  004136  013700  002400                     MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1472  004142  170127  040200                     LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
1473  004146  172437  002426                     LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1474  004152  172537  002436                     LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1475  004156  172737  002416                     LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1476  004162  170127  007600                     LDFPS   #007600         ;TURN INTERRUPTS ON
1477  004166  012737  004174  001110               MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1478
1479                                        ;*****************************************************************
1480
1481  004174  172600                       ARET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
1482  004176  172201                             ADDD    AC1,AC2         ;ADD AC1 BY AC2
1483  004200  170237  002362                     STFPS   FPS             ;STORE FLOATING POINT STATUS
1484  004204  023737  002362  002400               CMP     FPS,$FPS        ;CHECK FPS
1485  004212  001403                             BEQ     AERR2           ;BRANCH IF OK
1486  004214  174237  002406                     STD     AC2,ANS1        ;SAVE FPU ANSWER
1487  004220  104017                             ERROR   17              ;FPS ERROR
1488
1489  004222  005737  002400               AERR2:  TST     $FPS            ;ERROR BIT SET ?
1490  004226  100014                             BPL     ATST2           ;NO, DONT GET FEC/FEA
1491  004230  170337  002364                     STST    FEC             ;YES, CHECK STATUS
1492
1493  004234  023737  002364  002402               CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1494  004242  001401                             BEQ     1$              ;BRANCH IF OK
1495  004244  104027                             ERROR   27              ;FEC IS WRONG
1496
1497  004246  023737  002366  002404  1$:        CMP     FEA,$FEA        ;CHECK FLOATING PC
1498  004254  001401                             BEQ     ATST2           ;BRANCH IF OK
1499  004256  104027                             ERROR   27              ;WRONG ADDRESS IN FEA
1500
1501  004260  173702                       ATST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1502  004262  170000                             CFCC                    ;COPY FLOATING CONDITION CODES
1503  004264  001422                             BEQ     AEND2           ;ANSWERS CHECK
1504                                              ;COMPENSATE FOR FORTRAN INACCURACIES.
1505  004266  174237  002406                     STD     AC2,ANS1        ;SAVE FPU ANSWER
1506  004272  162737  000001  002414               SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
1507  004300  005637  002412                     SBC     ANS1+4
```

```
 1508  004304  005637  002410                  SBC     ANS1+2
 1509  004310  005637  002406                  SBC     ANS1
 1510  004314  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1511  004320  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1512  004322  001403                          BEQ     AEND2           ;BRANCH IF OK
 1513  004324  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 1514  004330  104006                          ERROR   6               ;FPU AND FORTRAN DISAGREE
 1515
 1516  004332  005037  002362          AEND2:  CLR     FPS             ;CLR FPU FPS BUFFER
 1517
```

```
                                        ;*****************************************************************
                                        ;*TEST 3        EXERCISE ADDF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;*****************************************************************
004336  000004                  TST3:   SCOPE
004340  012737  004472  002376          MOV     #ARET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
004346  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
004354  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
004360  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
004364  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
004370  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
004374  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
004400  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
004404  002426  002436                  .WORD   LONUM,HINUM     ;
004410  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
004414  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
004420  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
004424  023640                          $ADD                    ;ADDRESS OF FORTRAN ADD
004426  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
004432  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
004436  170127  040000                  LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
004442  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
004446  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
004452  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
004456  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
004462  012737  004470  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
004470  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
004472  172201                  ARET3:  ADDF    AC1,AC2         ;ADD AC1 BY AC2
004474  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
004500  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
004506  001403                          BEQ     ATST3           ;BRANCH IF OK
004510  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004514  104001                          ERROR   1               ;FPS ERROR
                                
004516  173702                  ATST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
004520  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
004522  001416                          BEQ     AEND3           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
004524  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004530  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
004536  005637  002406                  SBC     ANS1
004542  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
004546  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
004550  001403                          BEQ     AEND3           ;BRANCH IF OK
004552  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
004556  104005                          ERROR   5               ;FPU AND FORTRAN DISAGREE

004560  005037  002362          AEND3:  CLR     FPS             ;CLR FPU FPS BUFFER
```

```
                                        T4      EXERCISE ADDD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE

                                        ;;*****************************************************************
                                        ;*TEST 4        EXERCISE ADDD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
004564  000004                        TST4:   SCOPE
004566  012737  004720  002376              MOV     #ARET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
004574  012737  004600  002400              MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
004602  005037  002402                      CLR     $FEC            ;CLR FORTRAN FEC
004606  005037  002404                      CLR     $FEA            ;CLR FORTRAN FEA
004612  005037  002362                      CLR     FPS             ;CLR FPU FPS BUFFER
004616  005037  002364                      CLR     FEC             ;CLR FPU FEC BUFFER
004622  005037  002366                      CLR     FEA             ;CLR FPU FEA BUFFER
004626  004737  023404                      JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
004632  002426  002436                      .WORD   LONUM,HINUM     ;
004636  004437  023560                      JSR     R4,$POLSH       ;ENTER POLISH MODE
004642  023562  002426                      $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
004646  023562  002436                      $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
004652  023640                            $ADD                    ;ADDRESS OF FORTRAN ADD
004654  023612  002416                      $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

004660  013700  002400                      MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
004664  170127  040200                      LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
004670  172437  002426                      LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
004674  172537  002436                      LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
004700  172737  002416                      LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
004704  170127  004600                      LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
004710  012737  004716  001110              MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

004716  172600                        ARET4:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
004720  172201                              ADDD    AC1,AC2         ;ADD AC1 BY AC2
004722  170237  002362                      STFPS   FPS             ;STORE FLOATING POINT STATUS
004726  023737  002362  002400              CMP     FPS,$FPS        ;CHECK FPS
004734  001403                            BEQ     ATST4           ;BRANCH IF OK
004736  174237  002406                      STD     AC2,ANS1        ;SAVE FPU ANSWER
004742  104017                            ERROR   17              ;FPS ERROR

004744  173702                        ATST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
004746  170000                              CFCC                    ;COPY FLOATING CONDITION CODES
004750  001422                            BEQ     AEND4           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
004752  174237  002406                      STD     AC2,ANS1        ;SAVE FPU ANSWER
004756  162737  000001  002414              SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
004764  005637  002412                      SBC     ANS1+4
004770  005637  002410                      SBC     ANS1+2
004774  005637  002406                      SBC     ANS1
005000  173737  002406                      CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
005004  170000                              CFCC                    ;COPY FLOATING CONDITION CODES
005006  001403                            BEQ     AEND4           ;BRANCH IF OK
005010  174237  002406                      STD     AC2,ANS1        ;SAVE FPU ANSWER
005014  104006                            ERROR   6               ;FPU AND FORTRAN DISAGREE

005016  005037  002362              AEND4:  CLR     FPS             ;CLR FPU FPS BUFFER
```

```
                                ;;*****************************************************************
                                ;*TEST 5         EXERCISE ADDF, ALL INTERRUPTS ON, TRUNCATE MODE
                                ;;*****************************************************************
005022  000004                  TST5:   SCOPE
005024  012737  005156  002376          MOV     #ARET5,EXPFEA   ;ADDR OF INSTR BEING TESTED
005032  012737  007440  002400          MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
005040  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
005044  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
005050  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
005054  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
005060  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
005064  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
005070  002426  002436                  .WORD   LONUM,HINUM     ;
005074  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
005100  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
005104  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
005110  023640                          $ADD                    ;ADDRESS OF FORTRAN ADD
005112  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                
005116  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
005122  170127  040000                  LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
005126  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
005132  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
005136  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
005142  170127  007440                  LDFPS   #007440         ;TURN INTERRUPTS ON
005146  012737  005154  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                
                                ;*****************************************************************
                                
005154  172600                  ARET5:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
005156  172201                          ADDF    AC1,AC2         ;ADD AC1 BY AC2
005160  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005164  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
005172  001403                          BEQ     AERR5           ;BRANCH IF OK
005174  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
005200  104001                          ERROR   1               ;FPS ERROR
                                
005202  005737  002400          AERR5:  TST     $FPS            ;ERROR BIT SET ?
005206  100014                          BPL     ATST5           ;NO, DONT GET FEC/FEA
005210  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                
005214  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
005222  001401                          BEQ     1$              ;BRANCH IF OK
005224  104023                          ERROR   23              ;FEC IS WRONG
                                
005226  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
005234  001401                          BEQ     ATST5           ;BRANCH IF OK
005236  104023                          ERROR   23              ;WRONG ADDRESS IN FEA
                                
005240  173702                  ATST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
005242  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
005244  001427                          BEQ     AEND5           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
005246  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
005252  062737  000001  002410          ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
005260  005537  002406                  ADC     ANS1
```

```
 1680  005264  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1681  005270  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1682  005272  001414                          BEQ     AENDS           ;BRANCH IF OK
 1683  005274  162737  000002  002410          SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 1684  005302  005637  002406                  SBC     ANS1
 1685  005306  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1686  005312  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1687  005314  001403                          BEQ     AENDS           ;BRANCH IF OK
 1688  005316  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 1689  005322  104005                          ERROR   5               ;FPU AND FORTRAN DISAGREE
 1690
 1691  005324  005037  002362          AENDS:  CLR     FPS             ;CLR FPU FPS BUFFER
 1692
 1693
```

```
                                        ;;*****************************************************************
                                        ;*TEST 6         EXERCISE ADDD, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*****************************************************************
005330  001004                  TST6:   SCOPE
005332  012737  005464  002376          MOV     #ARET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
005340  012737  007640  002400          MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
005346  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
005352  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
005356  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
005362  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
005366  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
005372  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
005376  002426  002436                  .WORD   LONUM,HINUM     ;
005402  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
005406  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
005412  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
005416  023640                          $ADD                    ;ADDRESS OF FORTRAN ADD
005420  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

005424  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
005430  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
005434  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
005440  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
005444  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
005450  170127  007640                  LDFPS   #007640         ;TURN INTERRUPTS ON
005454  012737  005462  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

005462  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
005464  172201                  ARET6:  ADDD    AC1,AC2         ;ADD AC1 BY AC2
005466  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005472  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
005500  001403                          BEQ     AERR6           ;BRANCH IF OK
005502  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
005506  104017                          ERROR   17              ;FPS ERROR

005510  005737  002400          AERR6:  TST     $FPS            ;ERROR BIT SET ?
005514  100014                          BPL     ATST6           ;NO, DONT GET FEC/FEA
005516  170337  002364                  STST    FEC             ;YES, CHECK STATUS

005522  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
005530  001401                          BEQ     1$              ;BRANCH IF OK
005532  104027                          ERROR   27              ;FEC IS WRONG

005534  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
005542  001401                          BEQ     ATST6           ;BRANCH IF OK
005544  104027                          ERROR   27              ;WRONG ADDRESS IN FEA

005546  173702                  ATST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
005550  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
005552  001437                          BEQ     AEND6           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
005554  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
005560  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
005566  005537  002412                  ADC     ANS1+4
```

```
 1750  005572  005537  002410                  ADC     ANS1+2
 1751  005576  005537  002406                  ADC     ANS1
 1752  005602  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1753  005606  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1754  005610  001420                          BEQ     AEND6           ;BRANCH IF OK
 1755  005612  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 1756  005620  005637  002412                  SBC     ANS1+4
 1757  005624  005637  002410                  SBC     ANS1+2
 1758  005630  005637  002406                  SBC     ANS1
 1759  005634  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1760  005640  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 1761  005642  001403                          BEQ     AEND6           ;BRANCH IF OK
 1762  005644  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 1763  005650  104006                          ERROR   6               ;FPU AND FORTRAN DISAGREE
 1764
 1765  005652  005037  002362          AEND6:  CLR     FPS             ;CLR FPU FPS BUFFER
 1766
 1767
```

```
                                        ;;*****************************************************************
                                        ;*TEST 7        EXERCISE ADDF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                        ;;*****************************************************************
005656  000004                        T7:     SCOPE
005660  012737  006012  002376                MOV     #ARET7,EXPFEA   ;ADDR OF INSTR BEING TESTED
005666  012737  004440  002400                MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
005674  005037  002402                        CLR     $FEC            ;CLR FORTRAN FEC
005700  005037  002404                        CLR     $FEA            ;CLR FORTRAN FEA
005704  005037  002362                        CLR     FPS             ;CLR FPU FPS BUFFER
005710  005037  002364                        CLR     FEC             ;CLR FPU FEC BUFFER
005714  005037  002366                        CLR     FEA             ;CLR FPU FEA BUFFER
005720  004737  023414                        JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
005724  002426  002436                        .WORD   LONUM,HINUM     ;
005730  004437  023560                        JSR     R4,$POLSH       ;ENTER POLISH MODE
005734  023562  002426                        $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
005740  023562  002436                        $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
005744  023640                              $ADD                    ;ADDRESS OF FORTRAN ADD
005746  023612  002416                        $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
005752  013700  002400                        MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
005756  170127  040000                        LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
005762  172437  002426                        LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
005766  172537  002436                        LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
005772  172737  002416                        LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
005776  170127  004440                        LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
006002  012737  006010  001110                MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
006010  172600                          ARET7:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
006012  172201                                  ADDF    AC1,AC2         ;ADD AC1 BY AC2
006014  170237  002362                        STFPS   FPS             ;STORE FLOATING POINT STATUS
006020  023737  002362  002400                CMP     FPS,$FPS        ;CHECK FPS
006026  001403                                BEQ     ATST7           ;BRANCH IF OK
006030  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
006034  104001                                ERROR   1               ;FPS ERROR
006036  173702                          ATST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
006040  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
006042  001427                                BEQ     AEND7           ;ANSWERS CHECK
                                                ;COMPENSATE FOR FORTRAN INACCURACIES.
006044  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
006050  062737  000001  002410                ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
006056  005537  002406                        ADC     ANS1
006062  173737  002406                        CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
006066  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
006070  001414                                BEQ     AEND7           ;BRANCH IF OK
006072  162737  000002  002410                SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
006100  005637  002406                        SBC     ANS1
006104  173737  002406                        CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
006110  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
006112  001403                                BEQ     AEND7           ;BRANCH IF OK
006114  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
006120  104005                                ERROR   5               ;FPU AND FORTRAN DISAGREE
006122  005037  002362                  AEND7:  CLR     FPS             ;CLR FPU FPS BUFFER
```

```
1824
```

```
                                        ;;*****************************************************************
                                        ;*TEST 10       EXERCISE ADDD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                        ;;*****************************************************************
006126  000004                  TST10:  SCOPE
006130  012737  006262  002376          MOV     #ARET10,EXPFEA  ;ADDR OF INSTR BEING TESTED
006136  012737  004640  002400          MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
006144  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
006150  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
006154  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
006160  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
006164  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
006170  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
006174  002426  002436                  .WORD   LONUM,HINUM
006200  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
006204  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
006210  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
006214  023640                          $ADD                    ;ADDRESS OF FORTRAN ADD
006216  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

006222  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
006226  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
006232  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
006236  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
006242  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
006246  170127  004640                  LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
006252  012737  006260  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

006260  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
006262  172201                  ARET10: ADDD    AC1,AC2         ;ADD AC1 BY AC2
006264  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006270  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
006276  001403                          BEQ     ATST10          ;BRANCH IF OK
006300  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
006304  104017                          ERROR   17              ;FPS ERROR

006306  173702                  ATST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
006310  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
006312  001437                          BEQ     AEND10          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
006314  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
006320  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
006326  005537  002412                  ADC     ANS1+4
006332  005537  002410                  ADC     ANS1+2
006336  005537  002406                  ADC     ANS1
006342  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
006346  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
006350  001420                          BEQ     AEND10          ;BRANCH IF OK
006352  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
006360  005637  002412                  SBC     ANS1+4
006364  005637  002410                  SBC     ANS1+2
006370  005637  002406                  SBC     ANS1
006374  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
006400  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
006402  001403                          BEQ     AEND10          ;BRANCH IF OK
```

```
  1881  006404  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
  1882  006410  104006                          ERR^R   6               ;FPU AND FORTRAN DISAGREE
  1883
  1884  006412  005037  002362          AEND10: CLR     FPS             ;CLR FPU FPS BUFFER
  1885
```

```
1886                                    ;;*****************************************************************
1887                                    ;*TEST 11       EXERCISE SUBF, ALL INTERRUPTS ON, ROUNDING MODE
1888                                    ;;*****************************************************************
1889  006416  000004                    TST11:  SCOPE
1890  006420  012737  006552  002376            MOV     #ARET11,EXPFEA  ;ADDR OF INSTR BEING TESTED
1891  006426  012737  007400  002400            MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
1892  006434  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
1893  006440  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
1894  006444  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
1895  006450  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
1896  006454  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
1897  006460  004737  023414                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
1898  006464  002426  002436                    .WORD   LONUM,HINUM     ;
1899  006470  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
1900  006474  023562  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
1901  006500  023562  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
1902  006504  023634                            $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
1903  006506  023612  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
1904
1905  006512  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
1906  006516  170127  040000                    LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
1907  006522  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
1908  006526  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
1909  006532  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
1910  006536  170127  007400                    LDFPS   #007400         ;TURN INTERRUPTS ON
1911  006542  012737  006550  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
1912
1913                                            ;*****************************************************************
1914
1915  006550  172600                    ARET11: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
1916  006552  173201                            SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
1917  006554  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1918  006560  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
1919  006566  001403                            BEQ     AERR11          ;BRANCH IF OK
1920  006570  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
1921  006574  104002                            ERROR   2               ;FPS ERROR
1922
1923  006576  005737  002400            AERR11: TST     $FPS            ;ERROR BIT SET ?
1924  006602  100014                            BPL     ATST11          ;NO, DONT GET FEC/FEA
1925  006604  170337  002364                    STST    FEC             ;YES, CHECK STATUS
1926
1927  006610  023737  002364  002402            CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
1928  006616  001401                            BEQ     1$              ;BRANCH IF OK
1929  006620  104024                            ERROR   24              ;FEC IS WRONG
1930
1931  006622  023737  002366  002404    1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
1932  006630  001401                            BEQ     ATST11          ;BRANCH IF OK
1933  006632  104024                            ERROR   24              ;WRONG ADDRESS IN FEA
1934
1935  006634  173702                    ATST11: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
1936  006636  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
1937  006640  001416                            BEQ     AEND11          ;ANSWERS CHECK
1938                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
1939  006642  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
1940  006646  162737  000001  002410            SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
1941  006654  005637  002406                    SBC     ANS1
```

```
 1942  006660  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 1943  006664  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 1944  006666  001403                            BEQ     AEND11          ;BRANCH IF OK
 1945  006670  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
 1946  006674  104007                            ERROR   7               ;FPU AND FORTRAN DISAGREE
 1947
 1948  006676  005037  002362          AEND11: CLR     FPS             ;CLR FPU FPS BUFFER
 1949
```

```
                                        ;;*****************************************************************
                                        ;*TEST 12       EXERCISE SUBD, ALL INTERRUPTS ON, ROUNDING MODE
                                        ;;*****************************************************************
006702  000004                  TST12:  SCOPE
006704  012737  007036  002376          MOV     #ARET12,EXPFEA  ;ADDR OF INSTR BEING TESTED
006712  012737  007600  002400          MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
006720  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
006724  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
006730  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
006734  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
006740  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
006744  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
006750  002426  002436                  .WORD   LONUM,HINUM     ;
006754  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
006760  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
006764  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
006770  023634                          $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
006772  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

006776  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
007002  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
007006  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
007012  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
007016  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
007022  170127  007600                  LDFPS   #007600         ;TURN INTERRUPTS ON
007026  012737  007034  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

                                        ;*****************************************************************
007034  172600                  ARET12: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
007036  173201                          SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
007040  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007044  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
007052  001403                          BEQ     AERR12          ;BRANCH IF OK
007054  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
007060  104020                          ERROR   20              ;FPS ERROR

007062  005737  002400          AERR12: TST     $FPS            ;ERROR BIT SET ?
007066  100014                          BPL     ATST12          ;NO, DONT GET FEC/FEA
007070  170337  002364                  STST    FEC             ;YES, CHECK STATUS

007074  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
007102  001401                          BEQ     1$              ;BRANCH IF OK
007104  104030                          ERROR   30              ;FEC IS WRONG

007106  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
007114  001401                          BEQ     ATST12          ;BRANCH IF OK
007116  104030                          ERROR   30              ;WRONG ADDRESS IN FEA

007120  173702                  ATST12: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
007122  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
007124  001422                          BEQ     AEND12          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
007126  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
007132  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
007140  005637  002412                  SBC     ANS1+4
```

```
 2006  007144  005637  002410                  SBC     ANS1+2
 2007  007150  005637  002406                  SBC     ANS1
 2008  007154  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2009  007160  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2010  007162  001403                          BEQ     AEND12          ;BRANCH IF OK
 2011  007164  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 2012  007170  104010                          ERROR   10              ;FPU AND FORTRAN DISAGREE
 2013
 2014  007172  005037  002362          AEND12: CLR     FPS             ;CLR FPU FPS BUFFER
 2015
```

```
2016                                    ;;*****************************************************************
2017                                    ;*TEST 13       EXERCISE SUBF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
2018                                    ;;*****************************************************************
2019  007176  000004                    TST13:  SCOPE
2020  007200  012737  007332  002376            MOV     #ARET13,EXPFEA  ;ADDR OF INSTR BEING TESTED
2021  007206  012737  004400  002400            MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2022  007214  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
2023  007220  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
2024  007224  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
2025  007230  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
2026  007234  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
2027  007240  004737  023414                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2028  007244  002426  002436                    .WORD   LONUM,HINUM     ;
2029  007250  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
2030  007254  023562  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2031  007260  023562  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2032  007264  023634                            $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
2033  007266  023612  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2034
2035  007272  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2036  007276  170127  040000                    LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
2037  007302  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2038  007306  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2039  007312  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2040  007316  170127  004400                    LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
2041  007322  012737  007330  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2042
2043                                    ;*****************************************************************
2044
2045  007330  172600                    ARET13: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2046  007332  173201                            SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
2047  007334  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
2048  007340  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
2049  007346  001403                            BEQ     ATST13          ;BRANCH IF OK
2050  007350  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2051  007354  104002                            ERROR   2               ;FPS ERROR
2052
2053  007356  173702                    ATST13: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2054  007360  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2055  007362  001416                            BEQ     AEND13          ;ANSWERS CHECK
2056                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
2057  007364  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2058  007370  162737  000001  002410            SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
2059  007376  005637  002406                    SBC     ANS1
2060  007402  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2061  007406  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2062  007410  001403                            BEQ     AEND13          ;BRANCH IF OK
2063  007412  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2064  007416  104007                            ERROR   7               ;FPU AND FORTRAN DISAGREE
2065
2066  007420  005037  002362           AEND13: CLR     FPS             ;CLR FPU FPS BUFFER
2067
```

```
                                        ;;*****************************************************************
                                        ;*TEST 14       EXERCISE SUBD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
007424  000004                  TST14:  SCOPE
007426  012737  007560  002376          MOV     #ARET14,EXPFEA  ;ADDR OF INSTR BEING TESTED
007434  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
007442  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
007446  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
007452  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
007456  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
007462  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
007466  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
007472  002426  002436                  .WORD   LONUM,HINUM
007476  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
007502  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
007506  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
007512  023634                          $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
007514  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

007520  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
007524  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
007530  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
007534  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
007540  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
007544  170127  004600                  LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
007550  012737  007556  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

007556  172600                  ARET14: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
007560  173201                          SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
007562  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007566  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
007574  001403                          BEQ     ATST14          ;BRANCH IF OK
007576  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
007602  104020                          ERROR   20              ;FPS ERROR

007604  173702                  ATST14: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
007606  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
007610  001422                          BEQ     AEND14          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
007612  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
007616  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
007624  005637  002412                  SBC     ANS1+4
007630  005637  002410                  SBC     ANS1+2
007634  005637  002406                  SBC     ANS1
007640  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
007644  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
007646  001403                          BEQ     AEND14          ;BRANCH IF OK
007650  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
007654  104010                          ERROR   10              ;FPU AND FORTRAN DISAGREE

007656  005037  002362          AEND14: CLR     FPS             ;CLR FPU FPS BUFFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 15       EXERCISE SUBF, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*****************************************************************
007662  000004                        ST15:   SCOPE
007664  012737  010016  002376                MOV     #ARET15,EXPFEA  ;ADDR OF INSTR BEING TESTED
007672  012737  007440  002400                MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
007700  005037  002402                        CLR     $FEC            ;CLR FORTRAN FEC
007704  005037  002404                        CLR     $FEA            ;CLR FORTRAN FEA
007710  005037  002362                        CLR     FPS             ;CLR FPU FPS BUFFER
007714  005037  002364                        CLR     FEC             ;CLR FPU FEC BUFFER
007720  005037  002366                        CLR     FEA             ;CLR FPU FEA BUFFER
007724  004737  023414                        JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
007730  002426  002436                        .WORD   LONUM,HINUM     ;
007734  004437  023560                        JSR     R4,$POLSH       ;ENTER POLISH MODE
007740  023562  002426                        $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
007744  023562  002436                        $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
007750  023634                              $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
007752  023612  002416                        $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
007756  013700  002400                        MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
007762  170127  040000                        LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
007766  172437  002426                        LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
007772  172537  002436                        LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
007776  172737  002416                        LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
010002  170127  007440                        LDFPS   #007440         ;TURN INTERRUPTS ON
010006  012737  010014  001110                MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
010014  172600                          ARET15: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
010016  173201                                  SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
010020  170237  002362                        STFPS   FPS             ;STORE FLOATING POINT STATUS
010024  023737  002362  002400                CMF     FPS,$FPS        ;CHECK FPS
010032  001403                                BEQ     AERR15          ;BRANCH IF OK
010034  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
010040  104002                                ERROR   2               ;FPS ERROR
010042  005737  002400                AERR15: TST     $FPS            ;ERROR BIT SET ?
010046  100014                                BPL     ATST15          ;NO, DONT GET FEC/FEA
010050  170337  002364                        STST    FEC             ;YES, CHECK STATUS
010054  023737  002364  002402                CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
010062  001401                                BEQ     1$              ;BRANCH IF OK
010064  104024                                ERROR   24              ;FEC IS WRONG
010066  023737  002366  002404          1$:   CMP     FEA,$FEA        ;CHECK FLOATING PC
010074  001401                                BEQ     ATST15          ;BRANCH IF OK
010076  104024                                ERROR   24              ;WRONG ADDRESS IN FEA
010100  173702                        ATST15: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
010102  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
010104  001427                                BEQ     AEND15          ;ANSWERS CHECK
                                                ;COMPENSATE FOR FORTRAN INACCURACIES.
010106  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
010112  062737  000001  002410                ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
010120  005537  002406                        ADC     ANS1
```

```
 2178  010124  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2179  010130  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 2180  010132  001414                            BEQ     AEND15          ;BRANCH IF OK
 2181  010134  162737  000002  002410            SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 2182  010142  005637  002406                    SBC     ANS1
 2183  010146  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2184  010152  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 2185  010154  001403                            BEQ     AEND15          ;BRANCH IF OK
 2186  010156  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
 2187  010162  104007                            ERROR   7               ;FPU AND FORTRAN DISAGREE
 2188
 2189  010164  005037  002362          AEND15:   CLR     FPS             ;CLR FPU FPS BUFFER
 2190
```

```
                                        ;;*****************************************************************
                                        ;*TEST 16       EXERCISE SUBD, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*****************************************************************
010170  000004                  TST16:  SCOPE
010172  012737  010324  002376          MOV     #ARET16,EXPFEA  ;ADDR OF INSTR BEING TESTED
010200  012737  007640  002400          MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
010206  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
010212  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
010216  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
010222  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
010226  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
010232  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
010236  002426  002436                  .WORD   LONUM,HINUM     ;
010242  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
010246  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
010252  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
010256  023634                          $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
010260  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
                                        
010264  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
010270  170127  040200                  LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
010274  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
010300  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
010304  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
010310  170127  007640                  LDFPS   #007640         ;TURN INTERRUPTS ON
010314  012737  010322  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        
                                        ;*****************************************************************
                                        
010322  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
010324  173201                  ARET16: SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
010326  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
010332  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
010340  001403                          BEQ     AERR16          ;BRANCH IF OK
010342  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
010346  104020                          ERROR   20              ;FPS ERROR
                                        
010350  005737  002400          AERR16: TST     $FPS            ;ERROR BIT SET ?
010354  100014                          BPL     ATST16          ;NO, DONT GET FEC/FEA
010356  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                        
010362  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
010370  001401                          BEQ     1$              ;BRANCH IF OK
010372  104030                          ERROR   30              ;FEC IS WRONG
                                        
010374  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
010402  001401                          BEQ     ATST16          ;BRANCH IF OK
010404  104030                          ERROR   30              ;WRONG ADDRESS IN FEA
                                        
010406  173702                  ATST16: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
010410  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
010412  001437                          BEQ     AEND16          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
010414  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
010420  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
010426  005537  002412                  ADC     ANS1+4
```

```
 2247  010432  005537  002410                 ADC     ANS1+2
 2248  010436  005537  002406                 ADC     ANS1
 2249  010442  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2250  010446  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
 2251  010450  001420                         BEQ     AEND16          ;BRANCH IF OK
 2252  010452  162737  000002  002414         SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 2253  010460  005637  002412                 SBC     ANS1+4
 2254  010464  005637  002410                 SBC     ANS1+2
 2255  010470  005637  002406                 SBC     ANS1
 2256  010474  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2257  010500  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
 2258  010502  001403                         BEQ     AEND16          ;BRANCH IF OK
 2259  010504  174237  002406                 STD     AC2,ANS1        ;SAVE FPU ANSWER
 2260  010510  104010                         ERROR   10              ;FPU AND FORTRAN DISAGREE
 2261
 2262  010512  005037  002362         AEND16: CLR     FPS             ;CLR FPU FPS BUFFER
 2263
```

```
                                        ;;*****************************************************************
                                        ;*TEST 17       EXERCISE SUBF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                        ;;*****************************************************************
010516  000004                        TST17:  SCOPE
010520  012737  010652  002376                MOV     #ARET17,EXPFEA  ;ADDR OF INSTR BEING TESTED
010526  012737  004440  002400                MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
010534  005037  002402                        CLR     $FEC            ;CLR FORTRAN FEC
010540  005037  002404                        CLR     $FEA            ;CLR FORTRAN FEA
010544  005037  002362                        CLR     FPS             ;CLR FPU FPS BUFFER
010550  005037  002364                        CLR     FEC             ;CLR FPU FEC BUFFER
010554  005037  002366                        CLR     FEA             ;CLR FPU FEA BUFFER
010560  004737  023414                        JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
010564  002426  002436                        .WORD   LONUM,HINUM     ;
010570  004437  023560                        JSR     R4,$POLSH       ;ENTER POLISH MODE
010574  023562  002426                        $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
010600  023562  002436                        $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
010604  023634                              $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
010606  023612  002416                        $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                        
010612  013700  002400                        MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
010616  170127  040000                        LDFPS   #040000         ;LOAD FPS, INTERRUPT DISABLE
010622  172437  002426                        LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
010626  172537  002436                        LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
010632  172737  002416                        LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
010636  170127  004440                        LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
010642  012737  010650  001110                MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        
                                        ;*****************************************************************
                                        
010650  172600                        ARET17: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
010652  173201                                SUBF    AC1,AC2         ;SUBTRACT AC1 BY AC2
010654  170237  002362                        STFPS   FPS             ;STORE FLOATING POINT STATUS
010660  023737  002362  002400                CMP     FPS,$FPS        ;CHECK FPS
010666  001403                                BEQ     ATST17          ;BRANCH IF OK
010670  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
010674  104002                                ERROR   2               ;FPS ERROR
                                        
010676  173702                        ATST17: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
010700  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
010702  001427                                BEQ     AEND17          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
010704  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
010710  062737  000001  002410                ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
010716  005537  002406                        ADC     ANS1
010722  173737  002406                        CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
010726  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
010730  001414                                BEQ     AEND17          ;BRANCH IF OK
010732  162737  000002  002410                SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
010740  005637  002406                        SBC     ANS1
010744  173737  002406                        CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
010750  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
010752  001403                                BEQ     AEND17          ;BRANCH IF OK
010754  174237  002406                        STF     AC2,ANS1        ;SAVE FPU ANSWER
010760  104007                                ERROR   7               ;FPU AND FORTRAN DISAGREE
                                        
010762  005037  002362                AEND17: CLR     FPS             ;CLR FPU FPS BUFFER
```

```
2320
```

```
2321                                        ;;*****************************************************************
2322                                        ;*TEST 20       EXERCISE SUBD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
2323                                        ;;*****************************************************************
2324  010766  000004                        TST20:  SCOPE
2325  010770  012737  011122  002376                MOV     #ARET20,EXPFEA  ;ADDR OF INSTR BEING TESTED
2326  010776  012737  004640  002400                MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2327  011004  005037  002402                       CLR     $FEC            ;CLR FORTRAN FEC
2328  011010  005037  002404                       CLR     $FEA            ;CLR FORTRAN FEA
2329  011014  005037  002362                       CLR     FPS             ;CLR FPU FPS BUFFER
2330  011020  005037  002364                       CLR     FEC             ;CLR FPU FEC BUFFER
2331  011024  005037  002366                       CLR     FEA             ;CLR FPU FEA BUFFER
2332  011030  004737  023404                       JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2333  011034  002426  002436                       .WORD   LONUM,HINUM     ;
2334  011040  004437  023560                       JSR     R4,$POLSH       ;ENTER POLISH MODE
2335  011044  023562  002426                       $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2336  011050  023562  002436                       $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2337  011054  023634                             $SUB                    ;ADDRESS OF FORTRAN SUBTRACT
2338  011056  023612  002416                     $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2339
2340  011062  013700  002400                       MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2341  011066  170127  040200                       LDFPS   #040200         ;LOAD FPS, INTERRUPT DISABLE AND FD
2342  011072  172437  002426                       LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2343  011076  172537  002436                       LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2344  011102  172[illegible]  002416                LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2345  011106  170[illegible]  004640                LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
2346  011112  012[illegible]  011120  001110        MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2347
2348                                        ;*****************************************************************
2349
2350  011120  172600                        ARET20: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2351  011122  173201                                SUBD    AC1,AC2         ;SUBTRACT AC1 BY AC2
2352  011124  170237  002362                       STFPS   FPS             ;STORE FLOATING POINT STATUS
2353  011130  023737  002362  002400                CMP     FPS,$FPS        ;CHECK FPS
2354  011136  001403                               BEQ     ATST20          ;BRANCH IF OK
2355  011140  174237  002406                       STD     AC2,ANS1        ;SAVE FPU ANSWER
2356  011144  104020                               ERROR   20              ;FPS ERROR
2357
2358  011146  173702                        ATST20: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2359  011150  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
2360  011152  001437                               BEQ     AEND20          ;ANSWERS CHECK
2361                                              ;COMPENSATE FOR FORTRAN INACCURACIES.
2362  011154  174237  002406                       STD     AC2,ANS1        ;SAVE FPU ANSWER
2363  011160  062737  000001  002414                ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
2364  011166  005537  002412                       ADC     ANS1+4
2365  011172  005537  002410                       ADC     ANS1+2
2366  011176  005537  002406                       ADC     ANS1
2367  011202  173737  002406                       CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2368  011206  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
2369  011210  001420                               BEQ     AEND20          ;BRANCH IF OK
2370  011212  162737  000002  002414                SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
2371  011220  005637  002412                       SBC     ANS1+4
2372  011224  005637  002410                       SBC     ANS1+2
2373  011230  005637  002406                       SBC     ANS1
2374  011234  173737  002406                       CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
2375  011240  170000                                CFCC                    ;COPY FLOATING CONDITION CODES
2376  011242  001403                               BEQ     AEND20          ;BRANCH IF OK
```

```
 2377  011244  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 2378  011250  104010                          ERROR   10              ;FPU AND FORTRAN DISAGREE
 2379
 2380  011252  005037  002362          AEND20: CLR     FPS             ;CLR FPU FPS BUFFER
 2381
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21       EXERCISE MULF, ALL INTERRUPTS ON, ROUNDING MODE
                                        ;;*****************************************************************
011256  000004                  TST21:  SCOPE
011260  012737  011412  002376          MOV     #MRET1,EXPFEA   ;ADDR OF INSTR BEING TESTED
011266  012737  007400  002400          MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
011274  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
011300  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
011304  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
011310  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
011314  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
011320  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
011324  002426  002436                  .WORD   LONUM,HINUM     ;
011330  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
011334  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
011340  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
011344  025314                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
011346  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                        
011352  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
011356  170127  040000                  LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
011362  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
011366  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
011372  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
011376  170127  007400                  LDFPS   #007400         ;TURN INTERRUPTS ON
011402  012737  011410  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        
                                        ;*****************************************************************
                                        
011410  172600                  MRET1:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
011412  171201                          MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
011414  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
011420  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
011426  001403                          BEQ     MERR1           ;BRANCH IF OK
011430  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
011434  104003                          ERROR   3               ;FPS ERROR
                                        
011436  005737  002400          MERR1:  TST     $FPS            ;ERROR BIT SET ?
011442  100014                          BPL     MTST1           ;NO, DONT GET FEC/FEA
011444  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                        
011450  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
011456  001401                          BEQ     1$              ;BRANCH IF OK
011460  104025                          ERROR   25              ;FEC IS WRONG
                                        
011462  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
011470  001401                          BEQ     MTST1           ;BRANCH IF OK
011472  104025                          ERROR   25              ;WRONG ADDRESS IN FEA
                                        
011474  173702                  MTST1:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
011476  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
011500  001416                          BEQ     MEND1           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
011502  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
011506  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
011514  005637  002406                  SBC     ANS1
```

```
 2438  011520  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2439  011524  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2440  011526  001403                          BEQ     MEND1           ;BRANCH IF OK
 2441  011530  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 2442  011534  104011                          ERROR   11              ;FPU AND FORTRAN DISAGREE
 2443
 2444  011536  005037  002362          MEND1:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2445
```

```
2446                                    ;;*****************************************************************
2447                                    ;*TEST 22       EXERCISE MULD, ALL INTERRUPTS ON, ROUNDING MODE
2448                                    ;;*****************************************************************
2449  011542  000004                    TST22:  SCOPE
2450  011544  012737  011676  002376            MOV     #MRET2,EXPFEA   ;ADDR OF INSTR BEING TESTED
2451  011552  012737  007600  002400            MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2452  011560  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
2453  011564  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
2454  011570  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
2455  011574  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
2456  011600  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
2457  011604  004737  023404                    JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
2458  011610  002426  002436                    .WORD   LONUM,HINUM     ;
2459  011614  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
2460  011620  023562  002426                    $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
2461  011624  023562  002436                    $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
2462  011630  025314                            $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
2463  011632  023612  002416                    $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
2464
2465  011636  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2466  011642  170127  040200                    LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
2467  011646  172437  002426                    LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2468  011652  172537  002436                    LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2469  011656  172737  002416                    LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2470  011662  170127  007600                    LDFPS   #007600         ;TURN INTERRUPTS ON
2471  011666  012737  011674  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2472
2473                                    ;*****************************************************************
2474
2475  011674  172600                    MRET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
2476  011676  171201                            MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
2477  011700  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
2478  011704  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
2479  011712  001403                            BEQ     MERR2           ;BRANCH IF OK
2480  011714  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
2481  011720  104021                            ERROR   21              ;FPS ERROR
2482
2483  011722  005737  002400            MERR2:  TST     $FPS            ;ERROR BIT SET ?
2484  011726  100014                            BPL     MTST2           ;NO, DONT GET FEC/FEA
2485  011730  170337  002364                    STST    FEC             ;YES, CHECK STATUS
2486
2487  011734  023737  002364  002402            CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
2488  011742  001401                            BEQ     1$              ;BRANCH IF OK
2489  011744  104031                            ERROR   31              ;FEC IS WRONG
2490
2491  011746  023737  002366  002404    1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
2492  011754  001401                            BEQ     MTST2           ;BRANCH IF OK
2493  011756  104031                            ERROR   31              ;WRONG ADDRESS IN FEA
2494
2495  011760  173702                    MTST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2496  011762  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2497  011764  001422                            BEQ     MEND2           ;ANSWERS CHECK
2498                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
2499  011766  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
2500  011772  162737  000001  002414            SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
2501  012000  005637  002412                    SBC     ANS1+4
```

```
 2502  012004  005637  002410                  SBC     ANS1+2
 2503  012010  005637  002406                  SBC     ANS1
 2504  012014  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2505  012020  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2506  012022  001403                          BEQ     MEND2           ;BRANCH IF OK
 2507  012024  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 2508  012030  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
 2509
 2510  012032  005037  002362          MEND2:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2511
```

```
                                        ;;*****************************************************************
                                        ;*TEST 23       EXERCISE MULF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
012036  000004                  TST23:  SCOPE
012040  012737  012172  002376          MOV     #MRET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
012046  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
012054  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
012060  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
012064  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
012070  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
012074  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
012100  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
012104  002426  002436                  .WORD   LONUM,HINUM     ;
012110  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
012114  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
012120  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
012124  025314                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
012126  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
012132  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
012136  170127  040000                  LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
012142  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
012146  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
012152  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
012156  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
012162  012737  012170  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
012170  172600                  MRET3:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
012172  171201                          MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
012174  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
012200  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
012206  001403                          BEQ     MTST3           ;BRANCH IF OK
012210  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
012214  104003                          ERROR   3               ;FPS ERROR
012216  173702                  MTST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
012220  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
012222  001416                          BEQ     MEND3           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
012224  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
012230  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
012236  005637  002406                  SBC     ANS1
012242  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
012246  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
012250  001403                          BEQ     MEND3           ;BRANCH IF OK
012252  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
012256  104011                          ERROR   11              ;FPU AND FORTRAN DISAGREE
012260  005037  002362          MEND3:  CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                        T24     EXERCISE MULD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE

                                        ;;*****************************************************************
                                        ;*TEST 24       EXERCISE MULD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
012264  000004                  TST24:  SCOPE
012266  012737  012420  002376          MOV     #MRET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
012274  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
012302  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
012306  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
012312  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
012316  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
012322  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
012326  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
012332  002426  002436                  .WORD   LONUM,HINUM     ;
012336  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
012342  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
012346  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
012352  025314                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
012354  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

012360  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
012364  170127  040200                  LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
012370  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
012374  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
012400  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
012404  170127  004600                  LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
012410  012737  012416  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

012416  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
012420  171201                  MRET4:  MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
012422  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
012426  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
012434  001403                          BEQ     MTST4           ;BRANCH IF OK
012436  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
012442  104021                          ERROR   21              ;FPS ERROR

012444  173702                  MTST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
012446  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
012450  001422                          BEQ     MEND4           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
012452  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
012456  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
012464  005637  002412                  SBC     ANS1+4
012470  005637  002410                  SBC     ANS1+2
012474  005637  002406                  SBC     ANS1
012500  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
012504  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
012506  001403                          BEQ     MEND4           ;BRANCH IF OK
012510  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
012514  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE

012516  005037  002362          MEND4:  CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 25       EXERCISE MULF, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*****************************************************************
012522 000004                           TST25:  SCOPE
012524 012737 012656 002376                     MOV     #MRET5,EXPFEA   ;ADDR OF INSTR BEING TESTED
012532 012737 ..7440 002400                     MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
012540 005037 002402                            CLR     $FEC            ;CLR FORTRAN FEC
012544 005037 002404                            CLR     $FEA            ;CLR FORTRAN FEA
012550 005037 002362                            CLR     FPS             ;CLR FPU FPS BUFFER
012554 005037 002364                            CLR     FEC             ;CLR FPU FEC BUFFER
012560 005037 002366                            CLR     FEA             ;CLR FPU FEA BUFFER
012564 004737 023414                            JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
012570 002426 002436                            .WORD   LONUM,HINUM     ;
012574 004437 023560                            JSR     R4,$POLSH       ;ENTER POLISH MODE
012600 023562 002426                            $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
012604 023562 002436                            $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
012610 025314                                   $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
012612 023612 002416                            $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                        ;
012616 013700 002400                            MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
012622 170127 040000                            LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
012626 172437 002426                            LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
012632 172537 002436                            LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
012636 172737 002416                            LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
012642 170127 007440                            LDFPS   #007440         ;TURN INTERRUPTS ON
012646 012737 012654 001110                     MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;
                                        ;*****************************************************************
                                        ;
012654 172600                           MRET5:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
012656 171201                                   MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
012660 170237 002362                            STFPS   FPS             ;STORE FLOATING POINT STATUS
012664 023737 002362 002400                     CMP     FPS,$FPS        ;CHECK FPS
012672 001403                                   BEQ     MERR5           ;BRANCH IF OK
012674 174237 002406                            STF     AC2,ANS1        ;SAVE FPU ANSWER
012700 104003                                   ERROR   3               ;FPS ERROR
                                        ;
012702 005737 002400                    MERR5:  TST     $FPS            ;ERROR BIT SET ?
012706 100014                                   BPL     MTST5           ;NO, DONT GET FEC/FEA
012710 170337 002364                            STST    FEC             ;YES, CHECK STATUS
                                        ;
012714 023737 002364 002402                     CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
012722 001401                                   BEQ     1$              ;BRANCH IF OK
012724 104025                                   ERROR   25              ;FEC IS WRONG
                                        ;
012726 023737 002366 002404             1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
012734 001401                                   BEQ     MTST5           ;BRANCH IF OK
012736 104025                                   ERROR   25              ;WRONG ADDRESS IN FEA
                                        ;
012740 173702                           MTST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
012742 170000                                   CFCC                    ;COPY FLOATING CONDITION CODES
012744 001427                                   BEQ     MEND5           ;ANSWERS CHECK
                                                ;COMPENSATE FOR FORTRAN INACCURACIES.
012746 174237 002406                            STF     AC2,ANS1        ;SAVE FPU ANSWER
012752 062737 000001 002410                     ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
012760 005537 002406                            ADC     ANS1
```

```
 2673  012764  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2674  012770  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2675  012772  001414                          BEQ     MENDS           ;BRANCH IF OK
 2676  012774  162737  000002  002410          SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 2677  013002  005637  002406                  SBC     ANS1
 2678  013006  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2679  013012  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2680  013014  001403                          BEQ     MENDS           ;BRANCH IF OK
 2681  013016  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 2682  013022  104011                          ERROR   11              ;FPU AND FORTRAN DISAGREE
 2683
 2684  013024  005037  002362          MENDS:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2685
```

```
                                        ;;*****************************************************************
                                        ;*TEST 26       EXERCISE MULD, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*****************************************************************
013030  000004                          TST26:  SCOPE
013032  012737  013164  002376                  MOV     #MRET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
013040  012737  007640  002400                  MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
013046  005037  002402                          CLR     $FEC            ;CLR FORTRAN FEC
013052  005037  002404                          CLR     $FEA            ;CLR FORTRAN FEA
013056  005037  002362                          CLR     FPS             ;CLR FPU FPS BUFFER
013062  005037  002364                          CLR     FEC             ;CLR FPU FEC BUFFER
013066  005037  002366                          CLR     FEA             ;CLR FPU FEA BUFFER
013072  004737  023404                          JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
013076  002426  002436                          .WORD   LONUM,HINUM     ;
013102  004437  023560                          JSR     R4,$POLSH       ;ENTER POLISH MODE
013106  023562  002426                          $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
013112  023562  002436                          $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
013116  025314                                  $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
013120  023612  002416                          $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

013124  013700  002400                          MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
013130  170127  040200                          LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
013134  172437  002426                          LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
013140  172537  002436                          LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
013144  172737  002416                          LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
013150  170127  007640                          LDFPS   #007640         ;TURN INTERRUPTS ON
013154  012737  013162  001110                  MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

013162  172600                          MRET6:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
013164  171201                                  MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
013166  170237  002362                          STFPS   FPS             ;STORE FLOATING POINT STATUS
013172  023737  002362  002400                  CMP     FPS,$FPS        ;CHECK FPS
013200  001403                                  BEQ     MERR6           ;BRANCH IF OK
013202  174237  002406                          STD     AC2,ANS1        ;SAVE FPU ANSWER
013206  104021                                  ERROR   21              ;FPS ERROR

013210  005737  002400                  MERR6:  TST     $FPS            ;ERROR BIT SET ?
013214  100014                                  BPL     MTST6           ;NO, DONT GET FEC/FEA
013216  170337  002364                          STST    FEC             ;YES, CHECK STATUS

013222  023737  002364  002402                  CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
013230  001401                                  BEQ     1$              ;BRANCH IF OK
013232  104031                                  ERROR   31              ;FEC IS WRONG

013234  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
013242  001401                                  BEQ     MTST6           ;BRANCH IF OK
013244  104031                                  ERROR   31              ;WRONG ADDRESS IN FEA

013246  173702                          MTST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
013250  170000                                  CFCC                    ;COPY FLOATING CONDITION CODES
013252  001437                                  BEQ     MEND6           ;ANSWERS CHECK
                                                ;COMPENSATE FOR FORTRAN INACCURACIES.
013254  174237  002406                          STD     AC2,ANS1        ;SAVE FPU ANSWER
013260  062737  000001  002414                  ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
013266  005537  002412                          ADC     ANS1+4
```

```
 2742  013272  005537  002410                    ADC     ANS1+2
 2743  013276  005537  002406                    ADC     ANS1
 2744  013302  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2745  013306  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 2746  013310  001420                            BEQ     MEND6           ;BRANCH IF OK
 2747  013312  162737  000002  002414            SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 2748  013320  005637  002412                    SBC     ANS1+4
 2749  013324  005637  002410                    SBC     ANS1+2
 2750  013330  005637  002406                    SBC     ANS1
 2751  013334  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2752  013340  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 2753  013342  001403                            BEQ     MEND6           ;BRANCH IF OK
 2754  013344  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
 2755  013350  104012                            ERROR   12              ;FPU AND FORTRAN DISAGREE
 2756
 2757  013352  005037  002362            MEND6:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2758
```

```
2759                                    ;;*****************************************************************
2760                                    ;*TEST 27       EXERCISE MULF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
2761                                    ;;*****************************************************************
2762  013356  000004                    TST27:  SCOPE
2763  013360  012737  013512  002376            MOV     #MRET7,EXPFEA   ;ADDR OF INSTR BEING TESTED
2764  013366  012737  004440  002400            MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
2765  013374  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
2766  013400  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
2767  013404  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
2768  013410  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
2769  013414  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
2770  013420  004737  023414                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
2771  013424  002426  002436                    .WORD   LONUM,HINUM     ;
2772  013430  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
2773  013434  023562  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
2774  013440  023562  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
2775  013444  025314                            $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
2776  013446  023612  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
2777
2778  013452  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
2779  013456  170127  040000                    LDFPS   #040000         ;CLEAR THE FPS, INTERRUPT DISABLE
2780  013462  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
2781  013466  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
2782  013472  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
2783  013476  170127  004440                    LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
2784  013502  012737  013510  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
2785
2786                                    ;*****************************************************************
2787
2788  013510  172600                    MRET7:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
2789  013512  171201                            MULF    AC1,AC2         ;MULTIPLY AC1 BY AC2
2790  013514  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
2791  013520  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
2792  013526  001403                            BEQ     MTST7           ;BRANCH IF OK
2793  013530  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2794  013534  104003                            ERROR   3               ;FPS ERROR
2795
2796  013536  173702                    MTST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
2797  013540  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2798  013542  001427                            BEQ     MEND7           ;ANSWERS CHECK
2799                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
2800  013544  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2801  013550  062737  000001  002410            ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
2802  013556  005537  002406                    ADC     ANS1
2803  013562  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2804  013566  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2805  013570  001414                            BEQ     MEND7           ;BRANCH IF OK
2806  013572  162737  000002  002410            SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
2807  013600  005637  002406                    SBC     ANS1
2808  013604  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
2809  013610  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
2810  013612  001403                            BEQ     MEND7           ;BRANCH IF OK
2811  013614  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
2812  013620  104011                            ERROR   11              ;FPU AND FORTRAN DISAGREE
2813
2814  013622  005037  002362            MEND7:  CLR     FPS             ;CLEAR FPP FPS BUFFER
```

2815

```
                                        ;;*****************************************************************
                                        ;*TEST 30       EXERCISE MULD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                        ;;*****************************************************************
013626  000004                  TST30:  SCOPE
013630  012737  013762  002376          MOV     #MRET10,EXPFEA  ;ADDR OF INSTR BEING TESTED
013636  012737  004640  002400          MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
013644  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
013650  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
013654  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
013660  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
013664  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
013670  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
013674  002426  002436                  .WORD   LONUM,HINUM     ;
013700  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
013704  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
013710  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
013714  025314                          $MUL                    ;ADDRESS OF FORTRAN MULTIPLY
013716  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

013722  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
013726  170127  040200                  LDFPS   #040200         ;SET FD OF FPS ONLY, INTERRUPT DISABLE
013732  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
013736  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
013742  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
013746  170127  004640                  LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
013752  012737  013760  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

013760  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
013762  171201                  MRET10: MULD    AC1,AC2         ;MULTIPLY AC1 BY AC2
013764  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
013770  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
013776  001403                          BEQ     MTST10          ;BRANCH IF OK
014000  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
014004  104021                          ERROR   21              ;FPS ERROR

014006  173702                  MTST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
014010  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
014012  001437                          BEQ     MEND10          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
014014  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
014020  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
014026  005537  002412                  ADC     ANS1+4
014032  005537  002410                  ADC     ANS1+2
014036  005537  002406                  ADC     ANS1
014042  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
014046  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
014050  001420                          BEQ     MEND10          ;BRANCH IF OK
014052  162737  000002  002414          SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
014060  005637  002412                  SBC     ANS1+4
014064  005637  002410                  SBC     ANS1+2
014070  005637  002406                  SBC     ANS1
014074  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
014100  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
014102  001403                          BEQ     MEND10          ;BRANCH IF OK
```

```
 2872  014104  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 2873  014110  104012                          ERROR   12              ;FPU AND FORTRAN DISAGREE
 2874
 2875  014112  005037  002362          MEND10: CLR     FPS             ;CLEAR FPP FPS BUFFER
 2876
```

```
                                        ;;*****************************************************************
                                        ;*TEST 31       EXERCISE DIVF, ALL INTERRUPTS ON, ROUNDING MODE
                                        ;;*****************************************************************
014116  000004                  TST31:  SCOPE
014120  012737  014252  002376          MOV     #DRET1,EXPFEA   ;ADDR OF INSTR BEING TESTED
014126  012737  007400  002400          MOV     #007400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
014134  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
014140  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
014144  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
014150  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
014154  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
014160  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
014164  002426  002436                  .WORD   LONUM,HINUM     ;
014170  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
014174  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
014200  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
014204  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
014206  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
                                        
014212  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
014216  170127  040000                  LDFPS   #040000         ;SET INTERRUPT DISABLE
014222  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
014226  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
014232  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
014236  170127  007400                  LDFPS   #007400         ;TURN INTERRUPTS ON
014242  012737  014250  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        
                                        ;*****************************************************************
                                        
014250  172600                          LDF     AC0,AC2         ;LOAD AC0 INTO AC2
014252  174601                  DRET1:  DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
014254  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
014260  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
014266  001403                          BEQ     DERR1           ;BRANCH IF OK
014270  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
014274  104016                          ERROR   16              ;FPS ERROR
                                        
014276  005737  002400          DERR1:  TST     $FPS            ;ERROR BIT SET ?
014302  100014                          BPL     DTST1           ;NO, DONT GET FEC/FEA
014304  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                        
014310  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
014316  001401                          BEQ     1$              ;BRANCH IF OK
014320  104026                          ERROR   26              ;FEC IS WRONG
                                        
014322  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
014330  001401                          BEQ     DTST1           ;BRANCH IF OK
014332  104026                          ERROR   26              ;WRONG ADDRESS IN FEA
                                        
014334  173702                  DTST1:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
014336  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
014340  001416                          BEQ     DEND1           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
014342  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
014346  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
014354  005637  002406                  SBC     ANS1
```

```
 2933  014360  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 2934  014364  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 2935  014366  001403                          BEQ     DEND1           ;BRANCH IF OK
 2936  014370  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 2937  014374  104013                          ERROR   13              ;FPU AND FORTRAN DISAGREE
 2938
 2939  014376  005037  002362          DEND1:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 2940
```

```
                                        ;;*****************************************************************
                                        ;*TEST 32       EXERCISE DIVD, ALL INTERRUPTS ON, ROUNDING MODE
                                        ;;*****************************************************************
014402  000004                  TST32:  SCOPE
014404  012737  014536  002376          MOV     #DRET2,EXPFEA   ;ADDR OF INSTR BEING TESTED
014412  012737  007600  002400          MOV     #007600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
014420  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
014424  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
014430  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
014434  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
014440  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
014444  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
014450  002426  002436                  .WORD   LONUM,HINUM     ;
014454  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
014460  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
014464  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
014470  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
014472  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

014476  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
014502  170127  040200                  LDFPS   #040200         ;SET FID AND FD
014506  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
014512  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
014516  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
014522  170127  007600                  LDFPS   #007600         ;TURN INTERRUPTS ON
014526  012737  014534  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

014534  172600                  DRET2:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
014536  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
014540  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
014544  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
014552  001403                          BEQ     DERR2           ;BRANCH IF OK
014554  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
014560  104022                          ERROR   22              ;FPS ERROR

014562  005737  002400          DERR2:  TST     $FPS            ;ERROR BIT SET ?
014566  100014                          BPL     DTST2           ;NO, DONT GET FEC/FEA
014570  170337  002364                  STST    FEC             ;YES, CHECK STATUS

014574  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
014602  001401                          BEQ     1$              ;BRANCH IF OK
014604  104032                          ERROR   32              ;FEC IS WRONG

014606  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
014614  001401                          BEQ     DTST2           ;BRANCH IF OK
014616  104032                          ERROR   32              ;WRONG ADDRESS IN FEA

014620  173702                  DTST2:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
014622  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
014624  001422                          BEQ     DEND2           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
014626  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
014632  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
014640  005637  002412                  SBC     ANS1+4
```

```
 2997  014644  005637  002410                  SBC     ANS1+2
 2998  014650  005637  002406                  SBC     ANS1
 2999  014654  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3000  014660  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 3001  014662  001403                          BEQ     DEND2           ;BRANCH IF OK
 3002  014664  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 3003  014670  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
 3004
 3005  014672  005037  002362          DEND2:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 3006
```

```
                                        ;;*****************************************************************
                                        ;*TEST 33       EXERCISE DIVF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
014676  000004                  TST33:  SCOPE
014700  012737  015032  002376          MOV     #DRET3,EXPFEA   ;ADDR OF INSTR BEING TESTED
014706  012737  004400  002400          MOV     #004400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
014714  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
014720  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
014724  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
014730  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
014734  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
014740  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
014744  002426  002436                  .WORD   LONUM,HINUM     ;
014750  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
014754  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
014760  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
014764  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
014766  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
014772  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
014776  170127  040000                  LDFPS   #040000         ;SET FID
015002  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
015006  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
015012  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
015016  170127  004400                  LDFPS   #004400         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
015022  012737  015030  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
015030  172600                  DRET3:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
015032  174601                          DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
015034  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
015040  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
015046  001403                          BEQ     DERR3           ;BRANCH IF OK
015050  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
015054  104016                          ERROR   16              ;FPS ERROR
015056  005737  002400          DERR3:  TST     $FPS            ;ERROR BIT SET ?
015062  100014                          BPL     DTST3           ;NO, DONT GET FEC/FEA
015064  170337  002364                  STST    FEC             ;YES, CHECK STATUS
015070  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
015076  001401                          BEQ     1$              ;BRANCH IF OK
015100  104026                          ERROR   26              ;FEC IS WRONG
015102  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
015110  001401                          BEQ     DTST3           ;BRANCH IF OK
015112  104026                          ERROR   26              ;WRONG ADDRESS IN FEA
015114  173702                  DTST3:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
015116  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
015120  001416                          BEQ     DEND3           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
015122  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
015126  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
015134  005637  002406                  SBC     ANS1
```

```
 3063  015140  173737  002406                  CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3064  015144  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 3065  015146  001403                          BEQ     DEND3           ;BRANCH IF OK
 3066  015150  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
 3067  015154  104013                          ERROR   13              ;FPU AND FORTRAN DISAGREE
 3068
 3069  015156  005037  002362          DEND3:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 3070
```

```
                                        ;;*****************************************************************
                                        ;*TEST 34       EXERCISE DIVD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, ROUNDING MODE
                                        ;;*****************************************************************
015162  000004                  TST34:  SCOPE
015164  012737  015316  002376          MOV     #DRET4,EXPFEA   ;ADDR OF INSTR BEING TESTED
015172  012737  004600  002400          MOV     #004600,$FPS    ;SET IE BITS IN FORTRAN ANSWER
015200  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
015204  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
015210  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
015214  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
015220  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
015224  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
015230  002426  002436                  .WORD   LONUM,HINUM     ;
015234  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
015240  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
015244  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
015250  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
015252  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
                                        
015256  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
015262  170127  040200                  LDFPS   #040200         ;SET FID AND FD
015266  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
015272  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
015276  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
015302  170127  004600                  LDFPS   #004600         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
015306  012737  015314  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        
                                        ;*****************************************************************
                                        
015314  172600                  DRET4:  LDD     AC0,AC2         ;LOAD AC0 INTO AC2
015316  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
015320  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
015324  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
015332  001403                          BEQ     DERR4           ;BRANCH IF OK
015334  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
015340  104022                          ERROR   22              ;FPS ERROR
                                        
015342  005737  002400          DERR4:  TST     $FPS            ;ERROR BIT SET ?
015346  100014                          BPL     DTST4           ;NO, DONT GET FEC/FEA
015350  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                        
015354  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
015362  001401                          BEQ     1$              ;BRANCH IF OK
015364  104032                          ERROR   32              ;FEC IS WRONG
                                        
015366  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
015374  001401                          BEQ     DTST4           ;BRANCH IF OK
015376  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
                                        
015400  173702                  DTST4:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
015402  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
015404  001422                          BEQ     DEND4           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
015406  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
015412  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
015420  005637  002412                  SBC     ANS1+4
```

```
  3127  015424  005637  002410                  SBC     ANS1+2
  3128  015430  005637  002406                  SBC     ANS1
  3129  015434  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
  3130  015440  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
  3131  015442  001403                          BEQ     DEND4           ;BRANCH IF OK
  3132  015444  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
  3133  015450  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
  3134
  3135  015452  005037  002362          DEND4:  CLR     FPS             ;CLEAR FPP FPS BUFFER
  3136
```

```
3137                                    ;;*****************************************************************
3138                                    ;*TEST 35       EXERCISE DIVF, ALL INTERRUPTS ON, TRUNCATE MODE
3139                                    ;;*****************************************************************
3140  015456  000004                    T$T35:  SCOPE
3141  015460  012737  015612  002376            MOV     #DRET5,EXPFEA   ;ADDR OF INSTR BEING TESTED
3142  015466  012737  007440  002400            MOV     #007440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
3143  015474  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
3144  015500  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
3145  015504  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
3146  015510  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
3147  015514  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
3148  015520  004737  023414                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
3149  015524  002426  002436                    .WORD   LONUM,HINUM     ;
3150  015530  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
3151  015534  023562  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
3152  015540  023562  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
3153  015544  026434                            $DIV                    ;ADDRESS OF FORTRAN DIVIDE
3154  015546  023612  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
3155
3156  015552  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
3157  015556  170127  040000                    LDFPS   #040000         ;SET FID
3158  015562  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
3159  015566  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
3160  015572  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
3161  015576  170127  007440                    LDFPS   #007440         ;TURN INTERRUPTS ON
3162  015602  012737  015610  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
3163
3164                                    ;*****************************************************************
3165
3166  015610  172600                    DRET5:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
3167  015612  174601                            DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
3168  015614  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
3169  015620  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
3170  015626  001403                            BEQ     DERR5           ;BRANCH IF OK
3171  015630  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3172  015634  104016                            ERROR   16              ;FPS ERROR
3173
3174  015636  005737  002400            DERR5:  TST     $FPS            ;ERROR BIT SET ?
3175  015642  100014                            BPL     DTST5           ;NO, DONT GET FEC/FEA
3176  015644  170337  002364                    STST    FEC             ;YES, CHECK STATUS
3177
3178  015650  023737  002364  002402            CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
3179  015656  001401                            BEQ     1$              ;BRANCH IF OK
3180  015660  104026                            ERROR   26              ;FEC IS WRONG
3181
3182  015662  023737  002366  002404    1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
3183  015670  001401                            BEQ     DTST5           ;BRANCH IF OK
3184  015672  104026                            ERROR   26              ;WRONG ADDRESS IN FEA
3185
3186  015674  173702                    DTST5:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
3187  015676  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3188  015700  001427                            BEQ     DEND5           ;ANSWERS CHECK
3189                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
3190  015702  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3191  015706  062737  000001  002410            ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
3192  015714  005537  002406                    ADC     ANS1
```

```
 3193  015720  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3194  015724  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 3195  015726  001414                            BEQ     DENDS           ;BRANCH IF OK
 3196  015730  162737  000002  002410            SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
 3197  015736  005637  002406                    SBC     ANS1
 3198  015742  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3199  015746  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
 3200  015750  001403                            BEQ     DENDS           ;BRANCH IF OK
 3201  015752  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
 3202  015756  104013                            ERROR   13              ;FPU AND FORTRAN DISAGREE
 3203
 3204  015760  005037  002362          DENDS:    CLR     FPS             ;CLEAR FPP FPS BUFFER
 3205
```

```
                                        ;;*******************************************************
                                        ;*TEST 36       EXERCISE DIVD, ALL INTERRUPTS ON, TRUNCATE MODE
                                        ;;*******************************************************
015764  000004                  TST36:  SCOPE
015766  012737  016120  002376          MOV     #DRET6,EXPFEA   ;ADDR OF INSTR BEING TESTED
015774  012737  007640  002400          MOV     #007640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
016002  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
016006  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
016012  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
016016  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
016022  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
016026  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
016032  002426  002436                  .WORD   LONUM,HINUM     ;
016036  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
016042  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
016046  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
016052  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
016054  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
                                        ;
016060  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
016064  170127  040200                  LDFPS   #040200         ;SET FID AND FD
016070  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
016074  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
016100  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
016104  170127  007640                  LDFPS   #007640         ;TURN INTERRUPTS ON
016110  012737  016116  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;
                                        ;*******************************************************************
                                        ;
016116  172600                          LDD     AC0,AC2         ;LOAD AC0 INTO AC2
016120  174601                  DRET6:  DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
016122  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
016126  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
016134  001403                          BEQ     DERR6           ;BRANCH IF OK
016136  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
016142  104022                          ERROR   22              ;FPS ERROR
                                        ;
016144  005737  002400          DERR6:  TST     $FPS            ;ERROR BIT SET ?
016150  100014                          BPL     DTST6           ;NO, DONT GET FEC/FEA
016152  170337  002364                  STST    FEC             ;YES, CHECK STATUS
                                        ;
016156  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
016164  001401                          BEQ     1$              ;BRANCH IF OK
016166  104032                          ERROR   32              ;FEC IS WRONG
                                        ;
016170  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
016176  001401                          BEQ     DTST6           ;BRANCH IF OK
016200  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
                                        ;
016202  173702                  DTST6:  CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
016204  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
016206  001437                          BEQ     DEND6           ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
016210  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
016214  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
016222  005537  002412                  ADC     ANS1+4
```

```
                                        EXERCISE DIVD, ALL INTERRUPTS ON, TRUNCATE MODE

 3262  016226  005537  002410                 ADC     ANS1+2
 3263  016232  005537  002406                 ADC     ANS1
 3264  016236  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3265  016242  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
 3266  016244  001420                         BEQ     DEND6           ;BRANCH IF OK
 3267  016246  162737  000002  002414         SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
 3268  016254  005637  002412                 SBC     ANS1+4
 3269  016260  005637  002410                 SBC     ANS1+2
 3270  016264  005637  002406                 SBC     ANS1
 3271  016270  173737  002406                 CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3272  016274  170000                         CFCC                    ;COPY FLOATING CONDITION CODES
 3273  016276  001403                         BEQ     DEND6           ;BRANCH IF OK
 3274  016300  174237  002406                 STD     AC2,ANS1        ;SAVE FPU ANSWER
 3275  016304  104014                         ERROR   14              ;FPU AND FORTRAN DISAGREE
 3276
 3277  016306  005037  002362         DEND6:  CLR     FPS             ;CLEAR FPP FPS BUFFER
 3278
```

```
                                        T37     EXERCISE DIVF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE

                                        ;;*****************************************************************
                                        ;*TEST 37       EXERCISE DIVF, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
                                        ;;*****************************************************************
016312  000004                        TST37:  SCOPE
016314  012737  016446  002376              MOV     #DRET7,EXPFEA   ;ADDR OF INSTR BEING TESTED
016322  012737  004440  002400              MOV     #004440,$FPS    ;SET IE BITS IN FORTRAN ANSWER
016330  005037  002402                      CLR     $FEC            ;CLR FORTRAN FEC
016334  005037  002404                      CLR     $FEA            ;CLR FORTRAN FEA
016340  005037  002362                      CLR     FPS             ;CLR FPU FPS BUFFER
016344  005037  002364                      CLR     FEC             ;CLR FPU FEC BUFFER
016350  005037  002366                      CLR     FEA             ;CLR FPU FEA BUFFER
016354  004737  023414                      JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
016360  002426  002436                      .WORD   LONUM,HINUM     ;
016364  004437  023560                      JSR     R4,$POLSH       ;ENTER POLISH MODE
016370  023562  002426                      $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
016374  023562  002436                      $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
016400  026434                            $DIV                    ;ADDRESS OF FORTRAN DIVIDE
016402  023612  002416                      $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE

016406  013700  002400                      MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
016412  170127  040000                      LDFPS   #040000         ;SET FID
016416  172437  002426                      LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
016422  172537  002436                      LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
016426  172737  002416                      LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
016432  170127  004440                      LDFPS   #004440         ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
016436  012737  016444  001110              MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

016444  172600                        DRET7:  LDF     AC0,AC2         ;LOAD AC0 INTO AC2
016446  174601                                DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
016450  170237  002362                      STFPS   FPS             ;STORE FLOATING POINT STATUS
016454  023737  002362  002400              CMP     FPS,$FPS        ;CHECK FPS
016462  001403                              BEQ     DERR7           ;BRANCH IF OK
016464  174237  002406                      STF     AC2,ANS1        ;SAVE FPU ANSWER
016470  104016                              ERROR   16              ;FPS ERROR

016472  005737  002400                DERR7:  TST     $FPS            ;ERROR BIT SET ?
016476  100014                              BPL     DTST7           ;NO, DONT GET FEC/FEA
016500  170337  002364                      STST    FEC             ;YES, CHECK STATUS

016504  023737  002364  002402              CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
016512  001401                              BEQ     1$              ;BRANCH IF OK
016514  104026                              ERROR   26              ;FEC IS WRONG

016516  023737  002366  002404        1$:   CMP     FEA,$FEA        ;CHECK FLOATING PC
016524  001401                              BEQ     DTST7           ;BRANCH IF OK
016526  104026                              ERROR   26              ;WRONG ADDRESS IN FEA

016530  173702                        DTST7:  CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
016532  170000                              CFCC                    ;COPY FLOATING CONDITION CODES
016534  001427                              BEQ     DEND7           ;ANSWERS CHECK
                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
016536  174237  002406                      STF     AC2,ANS1        ;SAVE FPU ANSWER
016542  062737  000001  002410              ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
016550  005537  002406                      ADC     ANS1
```

```
3335  016554  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
3336  016560  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3337  016562  001414                            BEQ     DEND7           ;BRANCH IF OK
3338  016564  162737  000002  002410            SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
3339  016572  005637  002406                    SBC     ANS1
3340  016576  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
3341  016602  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3342  016604  001403                            BEQ     DEND7           ;BRANCH IF OK
3343  016606  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3344  016612  104013                            ERROR   13              ;FPU AND FORTRAN DISAGREE
3345
3346  016614  005037  002362          DEND7:   CLR     FPS             ;CLEAR FPP FPS BUFFER
3347
```

```
3348                                    ;;*****************************************************************
3349                                    ;*TEST 40       EXERCISE DIVD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE
3350                                    ;;*****************************************************************
3351  016620  000004                    TST40:  SCOPE
3352  016622  012737  016754  002376            MOV     #DRET10,EXPFEA  ;ADDR OF INSTR BEING TESTED
3353  016630  012737  004640  002400            MOV     #004640,$FPS    ;SET IE BITS IN FORTRAN ANSWER
3354  016636  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
3355  016642  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
3356  016646  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
3357  016652  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
3358  016656  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
3359  016662  004737  023404                    JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
3360  016666  002426  002436                    .WORD   LONUM,HINUM     ;
3361  016672  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
3362  016676  023562  002426                    $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
3363  016702  023562  002436                    $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
3364  016706  026434                            $DIV                    ;ADDRESS OF FORTRAN DIVIDE
3365  016710  023612  002416                    $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
3366
3367  016714  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
3368  016720  170127  040200                    LDFPS   #040200         ;SET FID AND FD
3369  016724  172437  002426                    LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
3370  016730  172537  002436                    LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
3371  016734  172737  002416                    LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
3372  016740  170127  004640                    LDFPS   #004640         ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
3373  016744  012737  016752  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
3374
3375                                    ;*****************************************************************
3376
3377  016752  172600                    DRET10: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
3378  016754  174601                            DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
3379  016756  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
3380  016762  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
3381  016770  001403                            BEQ     DERR10          ;BRANCH IF OK
3382  016772  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
3383  016776  104022                            ERROR   22              ;FPS ERROR
3384
3385  017000  005737  002400            DERR10: TST     $FPS            ;ERROR BIT SET ?
3386  017004  100014                            BPL     DTST10          ;NO, DONT GET FEC/FEA
3387  017006  170337  002364                    STST    FEC             ;YES, CHECK STATUS
3388
3389  017012  023737  002364  002402            CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
3390  017020  001401                            BEQ     1$              ;BRANCH IF OK
3391  017022  104032                            ERROR   32              ;FEC IS WRONG
3392
3393  017024  023737  002366  002404  1$:       CMP     FEA,$FEA        ;CHECK FLOATING PC
3394  017032  001443                            BEQ     DEND10          ;BRANCH IF OK
3395  017034  104032                            ERROR   32              ;WRONG ADDRESS IN FEA
3396
3397  017036  173702                    DTST10: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
3398  017040  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3399  017042  001437                            BEQ     DEND10          ;ANSWERS CHECK
3400                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
3401  017044  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
3402  017050  062737  000001  002414            ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
3403  017056  005537  002412                    ADC     ANS1+4
```

EXERCISE DIVD, OVERFLOW AND UNDERFLOW INTERRUPTS OFF, TRUNCATE MODE

```
3404  017062  005537  002410                    ADC     ANS1+2
3405  017066  005537  002406                    ADC     ANS1
3406  017072  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
3407  017076  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3408  017100  001420                            BEQ     DEND10          ;BRANCH IF OK
3409  017102  162737  000002  002414            SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
3410  017110  005637  002412                    SBC     ANS1+4
3411  017114  005637  002410                    SBC     ANS1+2
3412  017120  005637  002406                    SBC     ANS1
3413  017124  173737  002406                    CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
3414  017130  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3415  017132  001403                            BEQ     DEND10          ;BRANCH IF OK
3416  017134  174237  002406                    STD     AC2,ANS1        ;SAVE FPU ANSWER
3417  017140  104014                            ERROR   14              ;FPU AND FORTRAN DISAGREE
3418
3419  017142  005037  002362          DEND10:   CLR     FPS             ;CLEAR FPP FPS BUFFER
3420
```

```
                                        ;;*****************************************************************
                                        ;*TEST 41       EXERCISE DIVF, INTERRUPT DISABLE SET, ROUNDING MODE
                                        ;;*****************************************************************
017146  000004                  TST41:  SCOPE
017150  012737  017302  002376          MOV     #DRET11,EXPFEA  ;ADDR OF INSTR BEING TESTED
017156  012737  047400  002400          MOV     #047400,$FPS    ;SET IE BITS IN FORTRAN ANSWER
017164  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
017170  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
017174  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
017200  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
017204  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
017210  004737  023414                  JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
017214  002426  002436                  .WORD   LONUM,HINUM
017220  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
017224  023562  002426                  $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
017230  023562  002436                  $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
017234  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
017236  023612  002416                  $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE

017242  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
017246  170127  040000                  LDFPS   #040000         ;SET FID
017252  172437  002426                  LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
017256  172537  002436                  LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
017262  172737  002416                  LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
017266  170127  047400                  LDFPS   #047400         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
017272  012737  017300  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

017300  172600                  DRET11: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
017302  174601                          DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
017304  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
017310  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
017316  001403                          BEQ     DERR11          ;BRANCH IF OK
017320  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
017324  104016                          ERROR   16              ;FPS ERROR

017326  005737  002400          DERR11: TST     $FPS            ;ERROR BIT SET ?
017332  100014                          BPL     DTST11          ;NO, DONT GET FEC/FEA
017334  170337  002364                  STST    FEC             ;YES, CHECK STATUS

017340  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
017346  001401                          BEQ     1$              ;BRANCH IF OK
017350  104026                          ERROR   26              ;FEC IS WRONG

017352  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
017360  001401                          BEQ     DTST11          ;BRANCH IF OK
017362  104026                          ERROR   26              ;WRONG ADDRESS IN FEA

017364  173702                  DTST11: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
017366  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
017370  001416                          BEQ     DEND11          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
017372  174237  002406                  STF     AC2,ANS1        ;SAVE FPU ANSWER
017376  162737  000001  002410          SUB     #1,ANS1+2       ;DECREMENT FPU ANSWER
017404  005637  002406                  SBC     ANS1
```

```
  3477  017410  173737  002406                CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
  3478  017414  170000                        CFCC                    ;COPY FLOATING CONDITION CODES
  3479  017416  001403                        BEQ     DEND11          ;BRANCH IF OK
  3480  017420  174237  002406                STF     AC2,ANS1        ;SAVE FPU ANSWER
  3481  017424  104013                        ERROR   13              ;FPU AND FORTRAN DISAGREE
  3482
  3483  017426  005037  002362        DEND11: CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 42       EXERCISE DIVD, INTERRUPT DISABLE SET, ROUNDING MODE
                                        ;;*****************************************************************
017432  000004                  TST42:  SCOPE
017434  012737  017566  002376          MOV     #DRET12,EXPFEA  ;ADDR OF INSTR BEING TESTED
017442  012737  047600  002400          MOV     #047600,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
017450  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
017454  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
017460  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
017464  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
017470  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
017474  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
017500  002426  002436                  .WORD   LONUM,HINUM     ;
017504  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
017510  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
017514  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
017520  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
017522  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE

017526  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
017532  170127  040200                  LDFPS   #040200         ;SET FID AND FD
017536  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
017542  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
017546  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
017552  170127  047600                  LDFPS   #047600         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
017556  012737  017564  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

017564  172600                  DRET12: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
017566  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
017570  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
017574  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
017602  001403                          BEQ     DERR12          ;BRANCH IF OK
017604  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
017610  104022                          ERROR   22              ;FPS ERROR

017612  005737  002400          DERR12: TST     $FPS            ;ERROR BIT SET ?
017616  100014                          BPL     DTST12          ;NO, DONT GET FEC/FEA
017620  170337  002364                  STST    FEC             ;YES, CHECK STATUS

017624  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
017632  001401                          BEQ     1$              ;BRANCH IF OK
017634  104032                          ERROR   32              ;FEC IS WRONG

017636  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
017644  001401                          BEQ     DTST12          ;BRANCH IF OK
017646  104032                          ERROR   32              ;WRONG ADDRESS IN FEA

017650  173702                  DTST12: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
017652  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
017654  001422                          BEQ     DEND12          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
017656  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
017662  162737  000001  002414          SUB     #1,ANS1+6       ;DECREMENT FPU ANSWER
017670  005637  002412                  SBC     ANS1+4
```

```
 3540  017674  005637  002410                  SBC     ANS1+2
 3541  017700  005637  002406                  SBC     ANS1
 3542  017704  173737  002406                  CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
 3543  017710  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
 3544  017712  001403                          BEQ     DEND12          ;BRANCH IF OK
 3545  017714  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
 3546  017720  104014                          ERROR   14              ;FPU AND FORTRAN DISAGREE
 3547
 3548  017722  005037  002362          DEND12: CLR     FPS             ;CLEAR FPP FPS BUFFER
 3549
```

```
3550                                    ;;*****************************************************************
3551                                    ;*TEST 43       EXERCISE DIVF, INTERRUPT DISABLE SET, TRUNCATE MODE
3552                                    ;;*****************************************************************
3553  017726  000004                    TST43:  SCOPE
3554  017730  012737  020062  002376            MOV     #DRET13,EXPFEA  ;ADDR OF INSTR BEING TESTED
3555  017736  012737  047440  002400            MOV     #047440,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
3556  017744  005037  002402                    CLR     $FEC            ;CLR FORTRAN FEC
3557  017750  005037  002404                    CLR     $FEA            ;CLR FORTRAN FEA
3558  017754  005037  002362                    CLR     FPS             ;CLR FPU FPS BUFFER
3559  017760  005037  002364                    CLR     FEC             ;CLR FPU FEC BUFFER
3560  017764  005037  002366                    CLR     FEA             ;CLR FPU FEA BUFFER
3561  017770  004737  023414                    JSR     PC,RANDL2       ;GET RANDOM INPUT DATA
3562  017774  002426  002436                    .WORD   LONUM,HINUM     ;
3563  020000  004437  023560                    JSR     R4,$POLSH       ;ENTER POLISH MODE
3564  020004  023562  002426                    $PUSH   ,LONUM          ;PUSH 2 WORDS ON STACK (LONUM)
3565  020010  023562  002436                    $PUSH   ,HINUM          ;PUSH 2 WORDS ON STACK (HINUM)
3566  020014  026434                            $DIV                    ;ADDRESS OF FORTRAN DIVIDE
3567  020016  023612  002416                    $POPX   ,ANS2           ;POP 2 WORDS AND EXIT POLISH MODE
3568
3569  020022  013700  002400                    MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
3570  020026  170127  040000                    LDFPS   #040000         ;SET FID
3571  020032  172437  002426                    LDF     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
3572  020036  172537  002436                    LDF     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
3573  020042  172737  002416                    LDF     ANS2,AC3        ;LOAD AC3 WITH THE SUM
3574  020046  170127  047440                    LDFPS   #047440         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
3575  020052  012737  020060  001110            MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
3576
3577                                    ;*****************************************************************
3578
3579  020060  172600                    DRET13: LDF     AC0,AC2         ;LOAD AC0 INTO AC2
3580  020062  174601                            DIVF    AC1,AC2         ;DIVIDE AC1 INTO AC2
3581  020064  170237  002362                    STFPS   FPS             ;STORE FLOATING POINT STATUS
3582  020070  023737  002362  002400            CMP     FPS,$FPS        ;CHECK FPS
3583  020076  001403                            BEQ     DERR13          ;BRANCH IF OK
3584  020100  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3585  020104  104016                            ERROR   16              ;FPS ERROR
3586
3587  020106  005737  002400            DERR13: TST     $FPS            ;ERROR BIT SET ?
3588  020112  100014                            BPL     DTST13          ;NO, DONT GET FEC/FEA
3589  020114  170337  002364                    STST    FEC             ;YES, CHECK STATUS
3590
3591  020120  023737  002364  002402            CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
3592  020126  001401                            BEQ     1$              ;BRANCH IF OK
3593  020130  104026                            ERROR   26              ;FEC IS WRONG
3594
3595  020132  023737  002366  002404    1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
3596  020140  001401                            BEQ     DTST13          ;BRANCH IF OK
3597  020142  104026                            ERROR   26              ;WRONG ADDRESS IN FEA
3598
3599  020144  173702                    DTST13: CMPF    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
3600  020146  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3601  020150  001427                            BEQ     DEND13          ;ANSWERS CHECK
3602                                            ;COMPENSATE FOR FORTRAN INACCURACIES.
3603  020152  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3604  020156  062737  000001  002410            ADD     #1,ANS1+2       ;INCREMENT FPU ANSWER
3605  020164  005537  002406                    ADC     ANS1
```

```
3606  020170  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
3607  020174  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3608  020176  001414                            BEQ     DEND13          ;BRANCH IF OK
3609  020200  162737  000002  002410            SUB     #2,ANS1+2       ;DECREMENT FPU ANSWER
3610  020206  005637  002406                    SBC     ANS1
3611  020212  173737  002406                    CMPF    ANS1,AC3        ;CHECK ANSWERS AGAIN
3612  020216  170000                            CFCC                    ;COPY FLOATING CONDITION CODES
3613  020220  001403                            BEQ     DEND13          ;BRANCH IF OK
3614  020222  174237  002406                    STF     AC2,ANS1        ;SAVE FPU ANSWER
3615  020226  104013                            ERROR   13              ;FPU AND FORTRAN DISAGREE
3616
3617  020230  005037  002362          DEND13:   CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 44       EXERCISE DIVD, INTERRUPT DISABLE SET, TRUNCATE MODE
                                        ;;*****************************************************************
020234  000004                  TST44:  SCOPE
020236  012737  020370  002376          MOV     #DRET14,EXPFEA  ;ADDR OF INSTR BEING TESTED
020244  012737  047640  002400          MOV     #047640,$FPS    ;SET FID AND IE BITS IN FORTRAN ANSWER
020252  005037  002402                  CLR     $FEC            ;CLR FORTRAN FEC
020256  005037  002404                  CLR     $FEA            ;CLR FORTRAN FEA
020262  005037  002362                  CLR     FPS             ;CLR FPU FPS BUFFER
020266  005037  002364                  CLR     FEC             ;CLR FPU FEC BUFFER
020272  005037  002366                  CLR     FEA             ;CLR FPU FEA BUFFER
020276  004737  023404                  JSR     PC,RANDL4       ;GET RANDOM INPUT DATA
020302  002426  002436                  .WORD   LONUM,HINUM     ;
020306  004437  023560                  JSR     R4,$POLSH       ;ENTER POLISH MODE
020312  023562  002426                  $PUSH   ,LONUM          ;PUSH 4 WORDS ON STACK (LONUM)
020316  023562  002436                  $PUSH   ,HINUM          ;PUSH 4 WORDS ON STACK (HINUM)
020322  026434                          $DIV                    ;ADDRESS OF FORTRAN DIVIDE
020324  023612  002416                  $POPX   ,ANS2           ;POP 4 WORDS AND EXIT POLISH MODE
020330  013700  002400                  MOV     $FPS,R0         ;DISPLAY FLOATING POINT STATUS
020334  170127  040200                  LDFPS   #040200         ;SET FID AND FD
020340  172437  002426                  LDD     LONUM,AC0       ;LOAD AC0 WITH A RANDOM NUMBER
020344  172537  002436                  LDD     HINUM,AC1       ;LOAD AC1 WITH A RANDOM NUMBER
020350  172737  002416                  LDD     ANS2,AC3        ;LOAD AC3 WITH THE SUM
020354  170127  047640                  LDFPS   #047640         ;SET INTERRUPT DISABLE AND INTERRUPT BITS
020360  012737  020366  001110          MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS
                                        ;*****************************************************************
020366  172600                  DRET14: LDD     AC0,AC2         ;LOAD AC0 INTO AC2
020370  174601                          DIVD    AC1,AC2         ;DIVIDE AC1 INTO AC2
020372  170237  002362                  STFPS   FPS             ;STORE FLOATING POINT STATUS
020376  023737  002362  002400          CMP     FPS,$FPS        ;CHECK FPS
020404  001403                          BEQ     DERR14          ;BRANCH IF OK
020406  174237  002405                  STD     AC2,ANS1        ;SAVE FPU ANSWER
020412  104022                          ERROR   22              ;FPS ERROR
020414  005737  002400          DERR14: TST     $FPS            ;ERROR BIT SET ?
020420  100014                          BPL     DTST14          ;NO, DONT GET FEC/FEA
020422  170337  002364                  STST    FEC             ;YES, CHECK STATUS
020426  023737  002364  002402          CMP     FEC,$FEC        ;CHECK THE FLOATING EXCEPTION CODES
020434  001401                          BEQ     1$              ;BRANCH IF OK
020436  104032                          ERROR   32              ;FEC IS WRONG
020440  023737  002366  002404  1$:     CMP     FEA,$FEA        ;CHECK FLOATING PC
020446  001401                          BEQ     DTST14          ;BRANCH IF OK
020450  104032                          ERROR   32              ;WRONG ADDRESS IN FEA
020452  173702                  DTST14: CMPD    AC2,AC3         ;COMPARE FPU ANSWER TO FORTRAN ANSWER
020454  170000                          CFCC                    ;COPY FLOATING CONDITION CODES
020456  001437                          BEQ     DEND14          ;ANSWERS CHECK
                                        ;COMPENSATE FOR FORTRAN INACCURACIES.
020460  174237  002406                  STD     AC2,ANS1        ;SAVE FPU ANSWER
020464  062737  000001  002414          ADD     #1,ANS1+6       ;INCREMENT FPU ANSWER
020472  005537  002412                  ADC     ANS1+4
```

```
3674  020476  005537  002410                   ADC     ANS1+2
3675  020502  005537  002406                   ADC     ANS1
3676  020506  173737  002406                   CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
3677  020512  170000                           CFCC                    ;COPY FLOATING CONDITION CODES
3678  020514  001420                           BEQ     DEND14          ;BRANCH IF OK
3679  020516  162737  000002  002414           SUB     #2,ANS1+6       ;DECREMENT FPU ANSWER
3680  020524  005637  002412                   SBC     ANS1+4
3681  020530  005637  002410                   SBC     ANS1+2
3682  020534  005637  002406                   SBC     ANS1
3683  020540  173737  002406                   CMPD    ANS1,AC3        ;CHECK ANSWERS AGAIN
3684  020544  170000                           CFCC                    ;COPY FLOATING CONDITION CODES
3685  020546  001403                           BEQ     DEND14          ;BRANCH IF OK
3686  020550  174237  002406                   STD     AC2,ANS1        ;SAVE FPU ANSWER
3687  020554  104014                           ERROR   14              ;FPU AND FORTRAN DISAGREE
3688
3689  020556  005037  002362           DEND14: CLR     FPS             ;CLEAR FPP FPS BUFFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 45       ADDF, SUBF, MULF, DIVF EXERCISER
                                        ;;*****************************************************************
020562  000004                        TST45:  SCOPE
                                                ;*UNDERFLOW, OVERFLOW INTERRUPTS OFF; TRUNCATE MODE

020564  012737  000440  002400                 MOV     #440,$FPS       ;SOFTWARE STATUS
020572  005037  002402                         CLR     $FEC            ;CLEAR STATUS
020576  005037  002404                         CLR     $FEA            ;
020602  005037  002362                         CLR     FPS             ;
020606  005037  002364                         CLR     FEC             ;
020612  005037  002366                         CLR     FEA             ;

020616  004737  023414                         JSR     PC,RANDL2       ;GET 6 FLOAT RANDOM NUMBERS
020622  002446  002476                         .WORD   OP1,OP4         ;
020626  004737  023414                         JSR     PC,RANDL2       ;
020632  002456  002506                         .WORD   OP2,OP5         ;
020636  004737  023414                 1$:     JSR     PC,RANDL2       ;
020642  002466  002516                         .WORD   OP3,OP6         ;
020646  032737  077600  002516                 BIT     #077600,OP6     ;LET'S NEVER DIVIDE BY ZERO
020654  001770                               BEQ     1$              ;EXPO OF OP6 IS ZERO, SO GET ANOTHER

020656  004437  023560                         JSR     R4,$POLSH       ;ENTER POLISH MODE TO CALCULATE:
020662  023562  002446                         $PUSH   ,OP1            ;
020666  023562  002456                         $PUSH   ,OP2            ;
020672  023634                               $SUB                    ;  ANS2 = (OP1-OP2) * (OP3+OP4/OP6) * OP5
020674  023562  002466                         $PUSH   ,OP3            ;
020700  023562  002476                         $PUSH   ,OP4            ;
020704  023562  002516                         $PUSH   ,OP6            ;
020710  026434                               $DIV                    ;
020712  023640                               $ADD                    ;
020714  025314                               $MUL                    ;
020716  023562  002506                         $PUSH   ,OP5            ;
020722  025314                               $MUL                    ;
020724  023612  002416                         $POPX   ,ANS2           ;

020730  170127  040000                         LDFPS   #040000         ;NO CHECKS
020734  172437  002416                         LDF     ANS2,AC0        ;GET SOFTWARE ANSWER
020740  013700  002400                         MOV     $FPS,R0         ;DISPLAY $FPS
020744  012737  020752  001110                 MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

020752  170127  000440                         LDFPS   #000440         ;INITIAL FPS
020756  172537  002446                         LDF     OP1,AC1         ;AC1 <- OP1
020762  173137  002456                         SUBF    OP2,AC1         ;AC1 <- OP1-OP2
020766  172637  002476                         LDF     OP4,AC2         ;AC2 <- OP4
020772  174637  002516                         DIVF    OP6,AC2         ;AC2 <- OP4/OP6
020776  172237  002466                         ADDF    OP3,AC2         ;AC2 <- OP3+OP4/OP6
021002  171102                               MULF    AC2,AC1         ;AC1 <- (OP1-OP2)*(OP3+OP4/OP6)
021004  171137  002506                         MULF    OP5,AC1         ;AC1 <- (OP1-OP2)*(OP3+OP4/OP6)*OP5

021010  170237  002362                         STFPS   FPS             ;STORE STATUS AFTERWARD
021014  023737  002362  002400                 CMP     FPS,$FPS        ;CHECK STATUS
021022  001403                               BEQ     ETST1           ;BRANCH IF OK
021024  174137  002406                         STF     AC1,ANS1        ;SAVE FPU ANSWER
```

```
 3746  021030  104033                          ERROR   33              ;FPS ERROR
 3747
 3748  021032  173401                  ETST1:  CMPF    AC1,AC0         ;ANSWER OK ? (FPU:SOFTWARE)
 3749  021034  170000                          CFCC                    ;COPY CC-S
 3750  021036  001473                          BEQ     EEND1           ;ANSWER CHECKS
 3751
 3752                                          ;COMPENSATE IN LOB FOR INACCURACIES
 3753  021040  174137  002406                  STF     AC1,ANS1        ;FPU ANSWER
 3754  021044  062737  000002  002410          ADD     #2,ANS1+2       ;INC +2, KEEP CARRY
 3755  021052  005537  002406                  ADC     ANS1            ;
 3756  021056  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3757  021062  170000                          CFCC                    ;COPY CC-S
 3758  021064  001460                          BEQ     EEND1           ;ANSWER CHECKS
 3759
 3760  021066  162737  000001  002410          SUB     #1,ANS1+2       ;INC +1, KEEP CARRY
 3761  021074  005537  002406                  ADC     ANS1            ;
 3762  021100  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3763  021104  170000                          CFCC                    ;COPY CC-S
 3764  021106  001447                          BEQ     EEND1           ;ANSWER CHECKS
 3765
 3766  021110  162737  000002  002410          SUB     #2,ANS1+2       ;DEC -1, KEEP CARRY
 3767  021116  005637  002406                  SBC     ANS1            ;
 3768  021122  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3769  021126  170000                          CFCC                    ;COPY CC-S
 3770  021130  001436                          BEQ     EEND1           ;ANSWER CHECKS
 3771
 3772  021132  162737  000001  002410          SUB     #1,ANS1+2       ;DEC -2, KEEP CARRY
 3773  021140  005637  002406                  SBC     ANS1            ;
 3774  021144  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3775  021150  170000                          CFCC                    ;COPY CC-S
 3776  021152  001425                          BEQ     EEND1           ;ANSWER CHECKS
 3777
 3778  021154  162737  000001  002410          SUB     #1,ANS1+2       ;DEC -3, KEEP CARRY
 3779  021162  005637  002406                  SBC     ANS1            ;
 3780  021166  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3781  021172  170000                          CFCC                    ;COPY CC-S
 3782  021174  001414                          BEQ     EEND1           ;ANSWER CHECKS
 3783
 3784  021176  162737  000001  002410          SUB     #1,ANS1+2       ;DEC -4, KEEP CARRY
 3785  021204  005637  002406                  SBC     ANS1            ;
 3786  021210  173437  002406                  CMPF    ANS1,AC0        ;CHECK AGAIN
 3787  021214  170000                          CFCC                    ;COPY CC-S
 3788  021216  001403                          BEQ     EEND1           ;ANSWER CHECKS
 3789
 3790  021220  174137  002406                  STF     AC1,ANS1        ;FPU ANSWER
 3791  021224  104035                          ERROR   35              ;ANSWERS DON'T CHECK
 3792
 3793  021226  005037  002362          EEND1:  CLR     FPS             ;CLEAR BUFFER
 3794
```

```
                                        ;;*****************************************************************
                                        ;*TEST 46        ADDD, SUBD, MULD, DIVD EXERCISER
                                        ;;*****************************************************************
021232  000004                        TST46:  SCOPE
                                        ;*UNDERFLOW, OVERFLOW INTERRUPTS OFF; ROUND MODE

021234  012737  000600  002400                MOV     #600,$FPS       ;SOFTWARE STATUS
021242  005037  002402                        CLR     $FEC            ;CLEAR STATUS
021246  005037  002404                        CLR     $FEA            ;
021252  005037  002362                        CLR     FPS             ;
021256  005037  002364                        CLR     FEC             ;
021262  005037  002366                        CLR     FEA             ;

021266  004737  023404                        JSR     PC,RANDL4       ;GET 6 DOUBLE FLOAT RANDOM NUMBERS
021272  002446  002476                        .WORD   OP1,OP4         ;
021276  004737  023404                        JSR     PC,RANDL4       ;
021302  002466  002456                        .WORD   OP3,OP2         ;
021306  004737  023404                1$:     JSR     PC,RANDL4       ;
021312  002506  002516                        .WORD   OP5,OP6         ;
021316  032737  077600  002506                BIT     #077600,OP5     ;LET'S NEVER DIVIDE BY ZERO
021324  001770                                BEQ     1$              ;OP5 IS ZERO, TRY AGAIN
021326  032737  077600  002516                BIT     #077600,OP6     ;
021334  001764                                BEQ     1$              ;OP6 IS ZERO, TRY AGAIN

021336  004437  023560                        JSR     R4,$POLSH       ;ENTER POLISH MODE TO CALCULATE:
021342  023562  002446                        $PUSH   ,OP1            ;
021346  023562  002466                        $PUSH   ,OP3            ;          /OP1+OP3\    /OP2-OP4\
021352  023640                                $ADD                    ;  ANS2 = I-------I * I-------I
021354  023562  002506                        $PUSH   ,OP5            ;          \  OP5  /    \  OP6  /
021360  026434                                $DIV                    ;
021362  023562  002456                        $PUSH   ,OP2            ;
021366  023562  002476                        $PUSH   ,OP4            ;
021372  023634                                $SUB                    ;
021374  023562  002516                        $PUSH   ,OP6            ;
021400  026434                                $DIV                    ;
021402  025314                                $MUL                    ;
021404  023612  002416                        $POPX   ,ANS2           ;

021410  170127  040200                        LDFPS   #040200         ;NO CHECKS
021414  172437  002416                        LDD     ANS2,AC0        ;GET SOFTWARE ANSWER
021420  013700  002400                        MOV     $FPS,R0         ;DISPLAY $FPS
021424  012737  021432  001110                MOV     #.+6,$LPADR     ;RESET LOOP ADDRESS

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

021432  170127  000600                        LDFPS   #000600         ;INITIAL FPS
021436  172537  002446                        LDD     OP1,AC1         ;AC1 <- OP1
021442  172137  002466                        ADDD    OP3,AC1         ;AC1 <- OP1+OP3
021446  174537  002506                        DIVD    OP5,AC1         ;AC1 <- (OP1+OP3)/OP5
021452  172637  002456                        LDD     OP2,AC2         ;AC2 <- OP2
021456  173237  002476                        SUBD    OP4,AC2         ;AC2 <- OP2-OP4
021462  174637  002516                        DIVD    OP6,AC2         ;AC2 <- (OP2-OP4)/OP6
021466  171102                                MULD    AC2,AC1         ;AC1 <- (OP1+OP3)/OP5*(OP2-OP4)/OP6

021470  170237  002362                        STFPS   FPS             ;STORE STATUS AFTERWARD
021474  023737  002362  002400                CMP     FPS,$FPS        ;CHECK STATUS
```

```
 3851  021502  001403                          BEQ     ETST2           ;BRANCH IF OK
 3852  021504  174137  002406                  STD     AC1,ANS1        ;SAVE FPU ANSWER
 3853  021510  104034                          ERROR   34              ;FPS ERROR
 3854
 3855  021512  173401                  ETST2:  CMPD    AC1,AC0         ;ANSWER OK ? (FPU:SOFTWARE)
 3856  021514  170000                          CFCC                    ;COPY CC-S
 3857  021516  001523                          BEQ     EEND2           ;ANSWER CHECKS
 3858
 3859                                          ;COMPENSATE IN LOB FOR INACCURACIES
 3860  021520  174137  002406                  STD     AC1,ANS1        ;FPU ANSWER
 3861  021524  062737  000002  002414          ADD     #2,ANS1+6       ;INC +2, KEEP CARRY
 3862  021532  005537  002412                  ADC     ANS1+4          ;
 3863  021536  005537  002410                  ADC     ANS1+2          ;
 3864  021542  005537  002406                  ADC     ANS1            ;
 3865  021546  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3866  021552  170000                          CFCC                    ;COPY CC-S
 3867  021554  001504                          BEQ     EEND2           ;ANSWER CHECKS
 3868
 3869  021556  162737  000001  002414          SUB     #1,ANS1+6       ;INC +1, KEEP CARRY
 3870  021564  005537  002412                  ADC     ANS1+4          ;
 3871  021570  005537  002410                  ADC     ANS1+2          ;
 3872  021574  005537  002406                  ADC     ANS1            ;
 3873  021600  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3874  021604  170000                          CFCC                    ;COPY CC-S
 3875  021606  001467                          BEQ     EEND2           ;ANSWER CHECKS
 3876
 3877  021610  162737  000002  002414          SUB     #2,ANS1+6       ;DEC -1, KEEP CARRY
 3878  021616  005637  002412                  SBC     ANS1+4          ;
 3879  021622  005637  002410                  SBC     ANS1+2          ;
 3880  021626  005637  002406                  SBC     ANS1            ;
 3881  021632  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3882  021636  170000                          CFCC                    ;COPY CC-S
 3883  021640  001452                          BEQ     EEND2           ;ANSWER CHECKS
 3884
 3885  021642  162737  000001  002414          SUB     #1,ANS1+6       ;DEC -2, KEEP CARRY
 3886  021650  005637  002412                  SBC     ANS1+4          ;
 3887  021654  005637  002410                  SBC     ANS1+2          ;
 3888  021660  005637  002406                  SBC     ANS1            ;
 3889  021664  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3890  021670  170000                          CFCC                    ;COPY CC-S
 3891  021672  001435                          BEQ     EEND2           ;ANSWER CHECKS
 3892
 3893  021674  162737  000001  002414          SUB     #1,ANS1+6       ;DEC -3, KEEP CARRY
 3894  021702  005637  002412                  SBC     ANS1+4          ;
 3895  021706  005637  002410                  SBC     ANS1+2          ;
 3896  021712  005637  002406                  SBC     ANS1            ;
 3897  021716  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3898  021722  170000                          CFCC                    ;COPY CC-S
 3899  021724  001420                          BEQ     EEND2           ;ANSWER CHECKS
 3900
 3901  021726  162737  000001  002414          SUB     #1,ANS1+6       ;DEC -4, KEEP CARRY
 3902  021734  005637  002412                  SBC     ANS1+4          ;
 3903  021740  005637  002410                  SBC     ANS1+2          ;
 3904  021744  005637  002406                  SBC     ANS1            ;
 3905  021750  173437  002406                  CMPD    ANS1,AC0        ;CHECK AGAIN
 3906  021754  170000                          CFCC                    ;COPY CC-S
```

```
 3907  021756  001403                          BEQ     EEND2           ;ANSWER CHECKS
 3908
 3909  021760  174137  002406                  STD     AC1,ANS1        ;FPU ANSWER
 3910  021764  104036                          ERROR   36              ;ANSWERS DON'T CHECK
 3911
 3912  021766  005037  002362          EEND2:  CLR     FPS             ;CLEAR BUFFER
 3913
```

```
3914                                    ;;*****************************************************************
3915                                    .SBTTL  SUB PASS END CONTROL
3916
3917  021772  000004                            SCOPE           ;CHECK FOR TEST ITERATIONS HERE
3918
3919                                    ;IF TEST ONLY EITHER HFP OR WFP, ENTER "EOP" ROUTINE DIRECTLY
3920
3921                                    ; IF IN ALTERNATE HFP/WFP MODE,
3922                                    ;   COMPLEMENT FLAG(5), HFP ENABLE BIT,
3923                                    ;   ENTER EOP ROUTINE ONLY IF ABOUT TO TEST HFP NEXT,
3924                                    ;       TESTING SEQUENCE IS:  PASS#1  HFP SUB-PASS
3925                                    ;                             PASS#1  WFP SUB-PASS
3926                                    ;                             PASS#2  HFP SUB-PASS
3927                                    ;                               ...
3928
3929  021774  076600  000022                  MED     ,RWHAMI         ;GET WHAMI INTO R0
3930  022000  032700  000020                  BIT     #BIT04,R0       ;1=HFP PRESENT, 0=NONE
3931  022004  001423                          BEQ     $EOP            ;EXIT IF NONE
3932
3933  022006  032777  000002  157132          BIT     #SW01,@SWR      ;1=HFP OR WFP TEST ONLY
3934  022014  001017                          BNE     $EOP            ;0=ALTERNATE HFP AND WFP TESTS
3935
3936  022016  012701  010000                  MOV     #BIT12,R1       ;HFP PRESENT, AND IN ALTERNATE MODE;
3937  022022  076600  000144                  MED     ,RFLAG          ;SO READ FLAGS,
3938  022026  030100                          BIT     R1,R0           ;COMPLEMENT FLAG(5)=BIT12=HFP ENABLE FLAG
3939  022030  001402                          BEQ     1$              ;
3940  022032  040100                          BIC     R1,R0           ;CLEAR BIT 12
3941  022034  000401                          BR      2$              ;
3942  022036  050100                  1$:     BIS     R1,R0           ;SET BIT 12
3943  022040  076600  000344          2$:     MED     ,WFLAG          ;REWRITE FLAGS
3944
3945  022044  030100                          BIT     R1,R0           ;HFP OR WFP NEXT ?
3946  022046  001002                          BNE     $EOP            ;IF HFP AGAIN, START NEW PASS
3947  022050  000137  003452                  JMP     @#SUBPAS        ;IF WFP, NEXT SUBPASS
3948
3949
3950                                    ;;*****************************************************************
3951
3952                                    .SBTTL  END OF PASS ROUTINE (MODIFIED SYSMAC)
3953
3954                                    ;*INCREMENT THE PASS NUMBER ($PASS)
3955                                    ;*IF SW(10)=0, DING BELL ON PASS END
3956                                    ;*IF SW(12)=0, TYPE STATISTICS ON PASS END
3957                                    ;*IF THERE'S A MONITOR, GO TO IT
3958                                    ;*  ELSE JUMP TO NEWPAS
3959
3960  022054  000004                  $EOP:   SCOPE
3961  022056  005037  001104                  CLR     $ERFLG          ;ZERO ERROR FLAG
3962  022062  005037  001102                  CLR     $TSTNM          ;ZERO TEST NUMBER
3963  022066  005037  001166                  CLR     $TIMES          ;ZERO NUMBER OF ITERATIONS
3964  022072  005237  001210                  INC     $PASS           ;INCREMENT PASS COUNT,
3965  022076  042737  100000  001210          BIC     #100000,$PASS   ;  BUT NEVER LET IN GO NEGATIVE
3966  022104  005327                          DEC     (PC)+           ;PASS LOOP ?
3967  022106  000400                  $EOPCT: .WORD   400             ;
3968  022110  003027                          BGT     $DOAGN          ;YES
3969  022112  012737                          MOV     (PC)+,@(PC)+    ;RESTORE COUNTER
```

```
 3970  022114  000400                  $ENDCT: .WORD   400             ;
 3971  022116  022106                          $EOPCT                  ;
 3972  022120  032777  002000  157020          BIT     #SW10,@SWR      ;BELL ON PASS END ?
 3973  022126  001002                          BNE     1$              ;NO
 3974  022130  104401  001172                  TYPE    ,$BELL          ;YES
 3975  022134  032777  010000  157004  1$:     BIT     #SW12,@SWR      ;INHIBIT MESSAGE ?
 3976  022142  001002                          BNE     $GET42          ;YES
 3977  022144  004737  022174                  JSR     PC,STATS        ;TYPE STATISTICS
 3978  022150  013700  000042          $GET42: MOV     @#42,R0         ;GET MONITOR ADDRESS
 3979  022154  001405                          BEQ     $DOAGN          ;NO MONITOR
 3980  022156  000005                          RESET                   ;CLEAR WORLD
 3981
 3982  022160  004710                  $ENDAD: JSR     PC,(R0)         ;GO TO MONITOR
 3983  022162  000240                          NOP                     ;
 3984  022164  000240                          NOP                     ;RESERVED FOR ACT11
 3985  022166  000240                          NOP                     ;
 3986
 3987  022170  000137  003404          $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
                                        ;*
022174  010046                 STATS: MOV     R0,-(SP)         ;SAVE REGISTERS

022176  012700  022302                 MOV     #SVEC1,R0        ;FIRST DATA LINE ADDR VECTOR
022202  001404                         BEQ     1$               ;NONE IF ZERO
022204  104401  022371                 TYPE    ,SHDR1           ;HEADER
022210  004737  022254                 JSR     PC,10$           ;DATA LINE
022214  012700  022326         1$:     MOV     #SVEC2,R0        ;SECOND DATA LINE ADDR VECTOR
022220  001404                         BEQ     2$               ;NONE IF ZERO
022222  104401  022736                 TYPE    ,SHDR2           ;HEADER
022226  004737  022254                 JSR     PC,10$           ;DATA LINE
022232  012700  022344         2$:     MOV     #SVEC3,R0        ;THIRD DATA LINE ADDR VECTOR
022236  001404                         BEQ     3$               ;NONE IF ZERO
022240  104401  023127                 TYPE    ,SHDR3           ;HEADER
022244  004737  022254                 JSR     PC,10$           ;DATA LINE

022250  012600                 3$:     MOV     (SP)+,R0         ;RESTORE REGISTERS
022252  000207                         RTS     PC               ;EXIT

                                        ;*INTERNAL SUBR FOR DATA LINE TYPEOUT
022254  013046                 10$:    MOV     @(R0)+,-(SP)     ;MOVE NUMBER ON STACK, BUMP TO NEXT
022256  104403                         TYPOS                    ;TYPE OCTAL
022260     006                         .BYTE   6                ;MAX 6 DIGITS
022261     000                         .BYTE   0                ;SUPPRESS LEADING ZEROS
022262  005710                         TST     (R0)             ;BUMPED TO LAST VECTOR ?
022264  001403                         BEQ     11$              ;YES, ONTO NEXT LINE
022266  104401  022364                 TYPE    ,SHT             ;TYPE A <HT>
022272  000770                         BR      10$              ;CONTINUE WITH THIS LINE
022274  104401  022366         11$:    TYPE    ,SCRLF           ;TYPE A <CR><LF>
022300  000207                         RTS     PC               ;DONE

                                        ;*DATA VECTORS:
022302  002526  002540  002542 SVEC1:  .WORD   ADDC0,ADDC5,ADDC6,ADDC7,ADDC8,ADDC1,ADDC2,ADDC3,ADDC4,0
022310  002544  002546  002530
022316  002532  002534  002536
022324  000000
022326  002550  002554  002556 SVEC2:  .WORD   MULC0,MULC2,MULC3,MULC4,MULC5,MULC1,0
022334  002560  002562  002552
022342  000000
022344  002564  002572  002574 SVEC3:  .WORD   DIVC0,DIVC3,DIVC4,DIVC5,DIVC6,DIVC1,DIVC2,0
022352  002576  002600  002566
022360  002570  000000

                                        ;*HEADERS, ETC:
022364  000011                 SHT:    .ASCIZ  <11>             ;<HT>
022366  005015     000         SCRLF:  .ASCIZ  <15><12>         ;<CR><LF>
```

```
4044  022371  000015  005012          SHDR1:  .ASCII  <15><12><12>
4045  022374  025052  020052  052123          .ASCII  "*** STATISTICS ***"<15><12><12>
4046  022402  052101  051511  044524
4047  022410  051503  025040  025052
4048  022416  005015  000012
4049  022421  000050  047516  042524          .ASCII  "(NOTE - ALL NUMBERS ARE UNSIGNED OCTAL INTEGERS)"<15><12><12>
4050  022426  026440  040440  046114
4051  022434  047040  046525  042502
4052  022442  051522  040440  042522
4053  022450  052440  051516  043511
4054  022456  042516  020104  041517
4055  022464  040524  020114  047111
4056  022472  042524  042507  051522
4057  022500  006451  005012
4058  022504  042101  027504  052523          .ASCII  "ADD/SUB INSTRUCTIONS:"<15><12>
4059  022512  020102  047111  052123
4060  022520  052522  052103  047511
4061  022526  051516  006472  000012
4062  022533  000124  052117  046101          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---  OP A=0  OP A#0  OP A=0      NO"<1
4063  022540  026411  026455  053117
4064  022546  051105  046106  053517
4065  022554  026455  004455  026455
4066  022562  047125  042504  043122
4067  022570  047514  026527  026455
4068  022576  047411  020120  036501
4069  022604  004460  050117  040440
4070  022612  030043  047411  020120
4071  022620  036501  004460  020040
4072  022626  047040  006517  000012
4073  022633  000116  046525  042502          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE  OP B#0  OP B=0  OP B=0   SHIFT"<
4074  022640  004522  047524  040524
4075  022646  020114  027527  047105
4076  022654  041101  042514  052011
4077  022662  052117  046101  053440
4078  022670  042457  040516  046102
4079  022676  004505  050117  041040
4080  022704  030043  047411  020120
4081  022712  036502  004460  050117
4082  022720  041040  030075  020011
4083  022726  044123  043111  006524
4084  022734  000012
4085  022736  005015  052515  052114  SHDR2:  .ASCII  <15><12>"MULTIPLY INSTRUCTIONS:"<15><12>
4086  022744  050111  054514  044440
4087  022752  051516  051124  041525
4088  022760  044524  047117  035123
4089  022766  005015
4090  022770  047524  040524  004514          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---  A=0 AND/OR"<15><12>
4091  022776  026455  047455  042526
4092  023004  043122  047514  026527
4093  023012  026455  026411  052455
4094  023020  042116  051105  046106
4095  023026  053517  026455  004455
4096  023034  036501  020060  047101
4097  023042  027504  051117  005015
4098  023050  052516  041115  051105          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE      B=0"<15><12>
4099  023056  052011  052117  046101
```

```
4100  023064  053440  042457  040516
4101  023072  046102  004505  047524
4102  023100  040524  020114  027527
4103  023106  047105  041101  042514
4104  023114  020011  020040  041040
4105  023122  030075  005015     000
4106  023127     015  042012  053111  SHDR3:   .ASCII  <15><12>"DIVIDE INSTRUCTIONS:"<15><12>
4107  023134  042111  020105  047111
4108  023142  052123  052522  052103
4109  023150  047511  051516  006472
4110  023156     012
4111  023157     124  052117  046101          .ASCII  "TOTAL  ---OVERFLOW---  --UNDERFLOW---   NUMER   DENOM"<15><12>
4112  023164  026411  026455  053117
4113  023172  051105  046106  053517
4114  023200  026455  004455  026455
4115  023206  047125  042504  043122
4116  023214  047514  026527  026455
4117  023222  020011  052516  042515
4118  023230  004522  042040  047105
4119  023236  046517  005015
4120  023242  052516  041115  051105          .ASCIZ  "NUMBER TOTAL W/ENABLE  TOTAL W/ENABLE     A=0     B=0"<15><12>
4121  023250  052011  052117  046101
4122  023256  053440  042457  040516
4123  023264  046102  004505  047524
4124  023272  040524  020114  027527
4125  023300  047105  041101  042514
4126  023306  020011  020040  036501
4127  023314  004460  020040  041040
4128  023322  030075  005015     000
4129          023330                  .EVEN           ;BACK TO AN EVEN BOUNDARY
```

```
                                        FPP TRAP CATCHER

                                .SBTTL  FPP TRAP CATCHER

023330  012637  002370       FPPILT: MOV     (SP)+,FPPOPC    ;POP OLD PC FOR DISPLAY
023334  012637  002372               MOV     (SP)+,FPPOPS    ;POP OLD PS FOR DISPLAY
023340  170237  002362               STFPS   FPS             ;GET FPS
023344  170337  002364               STST    FEC             ;GET FEC/FEA
023350  005737  002362               TST     FPS             ;TEST ERROR BIT
023354  100005                       BPL     1$              ;OFF - NO ERROR BIT SET, BUT TRAPPED
                                                             ;ON  - IT SHOULD BE ON A TRAP
023356  032737  040000  002362       BIT     #040000,FPS     ;TEST INTERRUPT ENABLE BIT
023364  001001                       BNE     1$              ;ON  - INTR DISABLED, BUT TRAPPED
                                                             ;OFF - ABLE TO INTR, SO IGNORE IT,
023366  000401                       BR      2$              ;       AND SKIP THE ERROR
023370  104015                1$:    ERROR   15              ;SIGNAL UNEXPECTED FPP TRAP
023372  013746  002372        2$:    MOV     FPPOPS,-(SP)    ;PUSH PSW
023376  013746  002370               MOV     FPPOPC,-(SP)    ;PUSH PC
023402  000002                       RTI                     ;CONTINUE,  RECOVER AT LAST TRAP ONLY
```

```
                                        .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
                                        ;*
023404  010546          RANDL4: MOV     R5,-(SP)        ;SAVE R5
023406  012705  000004          MOV     #4,R5           ;4 WORDS AT EACH
023412  000403          BR      RAND
023414  010546          RANDL2: MOV     R5,-(SP)        ;SAVE R5
023416  012705  000002          MOV     #2,R5           ;2 WORDS AT EACH
023422  010446          RAND:   MOV     R4,-(SP)        ;SAVE REGISTERS
023424  010346                  MOV     R3,-(SP)        ;
023426  010246                  MOV     R2,-(SP)        ;
023430  010146                  MOV     R1,-(SP)        ;
023432  010046                  MOV     R0,-(SP)        ;
023434  005C46                  CLR     -(SP)           ;EXTRA REGISTER
023436  016600  000016          MOV     16(SP),R0       ;GET PC FOR RETURN
023442  012003                  MOV     (R0)+,R3        ;FIRST NUMBER DEST PTR
023444  012004                  MOV     (R0)+,R4        ;SECOND NUMBER DEST PTR
023446  010066  000016          MOV     R0,16(SP)       ;STORE NEW RETURN ADDRESS
023452  011300                  MOV     (R3),R0         ;R0 INITIAL NUMBER
023454  011401                  MOV     (R4),R1         ;R1 INITIAL NUMBER
023456  012702  000007  1$:     MOV     #7,R2           ;SHIFT COUNT
023462  005016                  CLR     (SP)            ;CLEAR LOB
023464  006300          2$:     ASL     R0              ;SHIFT R0 LEFT
023466  006101                  ROL     R1              ;  AND ROTATE CARRY INTO R1
023470  006116                  ROL     (SP)            ;  AND ROTATE CARRY INTO EXT
023472  077204                  SOB     R2,2$           ;7 SHIFTS
023474  061316                  ADD     (R3),(SP)       ;ADD # TO MAKE * 129
023476  005501                  ADC     R1              ;PROPOGATE CARRY
023500  061401                  ADD     (R4),R1         ;ADD # TO MAKE * 129
023502  005516                  ADC     (SP)            ;PROPOGATE CARRY
023504  062700  001057          ADD     #001057,R0      ;ADD LOW CONSTANT
023510  005501                  ADC     R1              ;PROPOGATE CARRY
023512  005516                  ADC     (SP)            ;PROPOGATE CARRY
023514  062701  047401          ADD     #047401,R1      ;ADD HIGH CONSTANT
023520  005516                  ADC     (SP)            ;PROPOGATE CARRY
023522  062716  000006          ADD     #000006,(SP)    ;ADD HIGHEST CONSTANT
023526  061600                  ADD     (SP),R0         ;REPRIME R0 WITH HIGHEST DIGIT
023530  005501                  ADC     R1              ;PROPOGATE CARRY
023532  010023                  MOV     R0,(R3)+        ;SAVE R0
023534  010124                  MOV     R1,(R4)+        ;SAVE R1
023536  077531                  SOB     R5,1$           ;LOOP FOR REQ'D NUMBER OF WORDS
023540  005726                  TST     (SP)+           ;POP TEMP REG
023542  012600                  MOV     (SP)+,R0
023544  012601                  MOV     (SP)+,R1        ;RESTORE REGISTERS
023546  012602                  MOV     (SP)+,R2        ;
023550  012603                  MOV     (SP)+,R3        ;
023552  012604                  MOV     (SP)+,R4        ;
023554  012605                  MOV     (SP)+,R5        ;
023556  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
023560  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
023562  012400          $PUSH:  MOV     (R4)+,R0        ;GET PTR TO SOURCE
023564  105737  002400          TSTB    $FPS            ;
023570  100004                  BPL     1$              ;
023572  016046  000006          MOV     6(R0),-(SP)     ;PUSH DOUBLE ON STACK
023576  016046  000004          MOV     4(R0),-(SP)     ;
023602  016046  000002  1$:     MOV     2(R0),-(SP)     ;PUSH FLOAT ON STACK
023606  011046                  MOV     (R0),-(SP)      ;
023610  000134                  JMP     @(R4)+          ;

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

POLISH EXPRESSION ROUTINES

```
                                        ;*2/4 WORDS DEPENDING UPON F/D MODE
4260  023612  012400                    $POPX:  MOV     (R4)+,R0        ;GET PTR TO DESTINATION
4261  023614  012620                            MOV     (SP)+,(R0)+     ;POP FLOAT FROM STACK
4262  023616  012620                            MOV     (SP)+,(R0)+     ;
4263  023620  105737  002400                    TSTB    $FPS            ;
4264  023624  100002                            BPL     1$              ;
4265  023626  012620                            MOV     (SP)+,(R0)+     ;POP DOUBLE FROM STACK
4266  023630  012620                            MOV     (SP)+,(R0)+     ;
4267  023632  000204                    1$:     RTS     R4              ;EXIT POLISH MODE
```

```
                        .SBTTL  FLOATING POINT SOFTWARE ROUTINES

                        ;*FLOATING POINT SOFTWARE ADD/SUBTRACT ROUTINES
                        ;*
                        ;*  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 SUM/DIFFERENCE.
                        ;*
                        ;*  EXAMPLE:    $PUSH                       $PUSH
                        ;*              ADDR(OPERAND A)             ADDR(OPERAND A)
                        ;*              $PUSH                       $PUSH
                        ;*              ADDR(OPERAND B)             ADDR(OPERAND B)
                        ;*              $ADD                        $SUB
                        ;*              $POPX                       $POPX
                        ;*              ADDR(RESULT)                ADDR(RESULT)
                        ;*                RESULT=A+B                  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".
                        ;*
                        ;*  ALL REGISTERS ARE PRESERVED UPON ENTRY, AND RESTORED UPON
                        ;*    EXIT.  THE ROUTINES ARE RE-ENTRANT.
                        ;*
```

```
4324                                            ;*  ADAPTED FROM PDP-11 FORTRAN SOFTWARE
4325                                            ;*    BY DONALD NORTH, SEPTEMBER, 1975.
4326                                            ;*
4327
4328  023634  062716  100000          $SUB:   ADD     #100000,(SP)    ;CHANGE SIGN OF TOP ITEM
4329  023640  005237  002526          $ADD:   INC     ADDCO           ;CTR:  TOTAL NUMBER OF ADD/SUB
4330  023644  042737  000017  002400          BIC     #17,$FPS        ;CLEAR STATUS BITS N Z V C
4331  023652  105737  002400                  TSTB    $FPS            ;TEST MODE
4332  023656  100402                          BMI     27$             ;D-MODE
4333  023660  004737  027672                  JSR     PC,$CONV        ;D-MODE: CONVERT 2 F OPDS TO D
4334  023664  010046                  27$:    MOV     R0,-(SP)        ;
4335  023666  010146                          MOV     R1,-(SP)        ;
4336  023670  010246                          MOV     R2,-(SP)        ;SAVE ALL REGISTERS
4337  023672  010346                          MOV     R3,-(SP)        ;
4338  023674  010446                          MOV     R4,-(SP)        ;
4339  023676  010546                          MOV     R5,-(SP)        ;
4340  023700  005046                          CLR     -(SP)           ;CLEAR SIGNS
4341  023702  005004                          CLR     R4              ;CLEAR EXPONENTS
4342  023704  005005                          CLR     R5              ;
4343  023706  006366  000024                  ASL     24(SP)          ;SHIFT OUT SIGN OF TOP ITEM
4344  023712  006166  000022                  ROL     22(SP)          ;
4345  023716  006166  000020                  ROL     20(SP)          ;
4346  023722  006166  000016                  ROL     16(SP)          ;SHIFT A1
4347  023726  156604  000017                  BISB    17(SP),R4       ;GET E1
4348  023732  001011                          BNE     31$             ;JUMP IF NON ZERO
4349  023734  005726                          TST     (SP)+           ;FLUSH SIGNS
4350  023736  032766  077600  000024          BIT     #077600,24(SP)  ;B=0, A=0 TOO ?
4351  023744  001456                          BEQ     32$             ;YES, BOTH ZERO
4352  023746  005237  002532                  INC     ADDC2           ;NO, CTR:  B=0,A#0
4353  023752  000137  024762                  JMP     3$              ;DONE
4354  023756  106116                  31$:    ROLB    (SP)            ;GET S1
4355  023760  006366  000034                  ASL     34(SP)          ;SHIFT OUT SIGN OF 2ND ITEM
4356  023764  006166  000032                  ROL     32(SP)          ;
4357  023770  006166  000030                  ROL     30(SP)          ;
4358  023774  006166  000026                  ROL     26(SP)          ;SHIFT A2
4359  024000  156605  000027                  BISB    27(SP),R5       ;GET E2
4360  024004  001042                          BNE     2$              ;JUMP IF NON ZERO
4361  024006  106016                          RORB    (SP)            ;RECONSTRUCT A1
4362  024010  006066  000016                  ROR     16(SP)          ;
4363  024014  006066  000020                  ROR     20(SP)          ;
4364  024020  006066  000022                  ROR     22(SP)          ;
4365  024024  006066  000024                  ROR     24(SP)          ;
4366  024030  016666  000016  000026          MOV     16(SP),26(SP)   ;FIRST ARG TO TOP OF STACK
4367  024036  016666  000020  000030          MOV     20(SP),30(SP)   ;
4368  024044  016666  000022  000032          MOV     22(SP),32(SP)   ;
4369  024052  016666  000024  000034          MOV     24(SP),34(SP)   ;
4370  024060  005726                          TST     (SP)+           ;FLUSH SIGNS
4371  024062  032766  077600  000024          BIT     #077600,24(SP)  ;A=0, B=0 TOO ?
4372  024070  001404                          BEQ     32$             ;YES, BOTH ZERO
4373  024072  005237  0025[illegible]          INC     ADDC1           ;NO, CTR:  A=0, B#0
4374  024076  000137  024[illegible]          JMP     3$              ;DONE
4375  024102  005237  002[illegible]  32$:    INC     ADDC3           ;CTR:  A=0, B=0
4376  024106  000137  024742                  JMP     29$             ;DONE, TRUE ZERO RESULT
4377  024112  106166  000001          2$:     ROLB    1(SP)           ;GET S2
4378  024116  112766  000001  000027          MOVB    #1,27(SP)       ;INSERT NORMAL BIT
4379  024124  112766  000001  000017          MOVB    #1,17(SP)       ;   "      "    "
```

```
 4380  024132  160405                          SUB     R4,R5           ;R5=E2-E1, R4=E1
 4381  024134  003011                          BGT     4$              ;JUMP IF E2>E1
 4382  024136  016600  000026                  MOV     26(SP),R0       ;R0=A2
 4383  024142  016601  000030                  MOV     30(SP),R1       ;R1=B2
 4384  024146  016602  000032                  MOV     32(SP),R2       ;R2=C2
 4385  024152  016603  000034                  MOV     34(SP),R3       ;R3=D2
 4386  024156  000427                          BR      5$              ;GO CHECK SIGNS
 4387  024160  060504                  4$:     ADD     R5,R4           ;R5=E2-E1, R4=E2,E2>E1
 4388  024162  016600  000016                  MOV     16(SP),R0       ;R0=A1
 4389  024166  016601  000020                  MOV     20(SP),R1       ;R1=B1
 4390  024172  016602  000022                  MOV     22(SP),R2       ;R2=C1
 4391  024176  016603  000024                  MOV     24(SP),R3       ;R3=D1
 4392  024202  016666  000026  000016          MOV     26(SP),16(SP)   ;
 4393  024210  016666  000030  000020          MOV     30(SP),20(SP)   ;
 4394  024216  016666  000032  000022          MOV     32(SP),22(SP)   ;
 4395  024224  016666  000034  000024          MOV     34(SP),24(SP)   ;
 4396  024232  000316                          SWAB    (SP)            ;EXCHANGE SIGNS
 4397  024234  005405                          NEG     R5              ;E1-E2
 4398  024236  126616  000001          5$:     CMPB    1(SP),(SP)      ;COMPARE SIGNS
 4399  024242  001412                          BEQ     6$              ;SAME, GO CHECK EXPONENT
 4400  024244  005403                          NEG     R3              ;NEGATE OPERAND
 4401  024246  005502                          ADC     R2              ;
 4402  024250  005501                          ADC     R1              ;
 4403  024252  005500                          ADC     R0              ;
 4404  024254  005402                          NEG     R2              ;
 4405  024256  005501                          ADC     R1              ;
 4406  024260  005500                          ADC     R0              ;
 4407  024262  005401                          NEG     R1              ;
 4408  024264  005500                          ADC     R0              ;
 4409  024266  005400                          NEG     R0              ;
 4410  024270  005705                  6$:     TST     R5              ;CHECK EXPONENTS
 4411  024272  001466                          BEQ     7$              ;JUMP IF E1=E2
 4412  024274  022705  177707                  CMP     #-57.,R5        ;ANY POINT IN SHIFTING?
 4413  024300  003413                          BLE     8$              ;YES
 4414  024302  016600  000016                  MOV     16(SP),R0       ;NO, ANSWER IS OPERAND
 4415  024306  016601  000020                  MOV     20(SP),R1       ;WITH LARGER EXPONENT
 4416  024312  016602  000022                  MOV     22(SP),R2       ;
 4417  024316  016603  000024                  MOV     24(SP),R3       ;
 4418  024322  005237  002536                  INC     ADDC4           ;CTR:  NO SHIFT
 4419  024326  000501                          BR      9$              ;
 4420  024330  022705  177770          8$:     CMP     #-8.,R5         ;CHECK # OF BITS TO SHIFT
 4421  024334  003437                          BLE     10$             ;JUMP IF LESS THAN 1/2 WORD
 4422  024336  005700                          TST     R0              ;
 4423  024340  006746                          SXT     -(SP)           ;EXTEND SIGN
 4424  024342  022705  177760          12$:    CMP     #-16.,R5        ;
 4425  024346  002411                          BLT     11$             ;JUMP IF LESS THAN 1 WORD
 4426  024350  010203                          MOV     R2,R3           ;SHIFT A WORD AT A TIME
 4427  024352  010102                          MOV     R1,R2           ;
 4428  024354  010001                          MOV     R0,R1           ;
 4429  024356  011600                          MOV     (SP),R0         ;USE EXTENSION
 4430  024360  062705  000020                  ADD     #16.,R5         ;ADJUST EXPONENT
 4431  024364  001366                          BNE     12$             ;TRY AGAIN
 4432  024366  005726                          TST     (SP)+           ;POP EXTENSION
 4433  024370  000427                          BR      7$              ;SHIFT DONE
 4434  024372  022705  177775          11$:    CMP     #-3,R5          ;JUMP IF LESS THAN 4 TO SHIFT
 4435  024376  003415                          BLE     13$             ;
```

```
4436  024400  010416                        MOV     R4,(SP)         ;SAVE EXP & SHIFT COUNT
4437  024402  010546                        MOV     R5,-(SP)        ;
4438  024404  010104                        MOV     R1,R4           ;SAVE R1
4439  024406  073005                        ASHC    R5,R0           ;SHIFT HIGH ORDER
4440  024410  010205                        MOV     R2,R5           ;SAVE R2
4441  024412  073416                        ASHC    (SP),R4         ;SHIFT IT
4442  024414  010204                        MOV     R2,R4           ;
4443  024416  010502                        MOV     R5,R2           ;R2 DONE
4444  024420  010305                        MOV     R3,R5           ;SET UP LOW ORDER
4445  024422  073426                        ASHC    (SP)+,R4        ;DO LOW ORDER
4446  024424  010503                        MOV     R5,R3           ;
4447  024426  012604                        MOV     (SP)+,R4        ;RESTORE EXP TO R4
4448  024430  000407                        BR      7$              ;
4449  024432  005726                13$:    TST     (SP)+           ;POP EXTENSION
4450  024434  006200                10$:    ASR     R0              ;SHIFT RIGHT
4451  024436  006001                        ROR     R1              ;
4452  024440  006002                        ROR     R2              ;
4453  024442  006003                        ROR     R3              ;
4454  024444  005205                        INC     R5              ;COUNT LOOP
4455  024446  002772                        BLT     10$             ;
4456  024450  066603  000024        7$:     ADD     24(SP),R3       ;FORM SUM
4457  024454  005502                        ADC     R2              ;
4458  024456  005501                        ADC     R1              ;
4459  024460  005500                        ADC     R0              ;
4460  024462  066602  000022                ADD     22(SP),R2       ;
4461  024466  005501                        ADC     R1              ;
4462  024470  005500                        ADC     R0              ;
4463  024472  066601  000020                ADD     20(SP),R1       ;
4464  024476  005500                        ADC     R0              ;
4465  024500  066600  000016                ADD     16(SP),R0       ;
4466  024504  126616  000001                CMPB    1(SP),(SP)      ;CHECK FOR UNEQUAL SIGNS
4467  024510  001161                        BNE     14$             ;CLEAN UP SUBTRACT
4468  024512  030027  001000                BIT     R0,#1000        ;
4469  024516  001405                        BEQ     9$              ;JUMP IF NO NORMAL BIT OVERFLOW
4470  024520  006200                        ASR     R0              ;
4471  024522  006001                        ROR     R1              ;
4472  024524  006002                        ROR     R2              ;
4473  024526  006003                        ROR     R3              ;
4474  024530  005204                        INC     R4              ;INCREASE EXP
4475  024532  000304                9$:     SWAB    R4              ;MOVE EXP LEFT
4476  024534  001425                        BEQ     16$             ;JUMP IF NO OVERFLOW
4477  024536  105004                        CLRB    R4              ;CLEAR OVERFLOWED BITS
4478  024540  052737  000002  002400        BIS     #02,$FPS        ;SET V BIT ON OVERFLOW
4479  024546  005237  002540                INC     ADDC5           ;CTR:  OVERFLOW
4480  024552  032737  001000  002400        BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
4481  024560  001464                        BEQ     34$             ;NO, ZERO RESULT
4482  024562  005237  002542                INC     ADDC6           ;CTR:  OVERFLOW, ENABLED
4483  024566  052737  100000  002400        BIS     #100000,$FPS    ;YES, SET ERROR BIT
4484  024574  012737  000010  002402        MOV     #10,$FEC        ;SET $FEC
4485  024602  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
4486  024610  150004                16$:    BISB    R0,R4           ;INSERT HIGH ORDER FRACTION
4487  024612  006026                        ROR     (SP)+           ;INSERT SIGN
4488  024614  006004                        ROR     R4              ;
4489  024616  006001                        ROR     R1              ;
4490  024620  006002                        ROR     R2              ;
4491  024622  006003                        ROR     R3              ;
```

```
4492  024624  005503                        ADC     R3              ;
4493  024626  005502                        ADC     R2              ;
4494  024630  005501                        ADC     R1              ;
4495  024632  005504                        ADC     R4              ;
4496  024634  103401                        BCS     20$             ;OVERFLOW ON ROUND ?
4497  024636  102024                        BVC     30$             ;
4498  024640  052737  000002  002400  20$:  BIS     #02,$FPS        ;YES - SET V BIT
4499  024646  005237  002540                INC     ADDC5           ;CTR:  OVERFLOW
4500  024652  032737  001000  002400        BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
4501  024660  001430                        BEQ     29$             ;NO, ZERO RESULT
4502  024662  005237  002542                INC     ADDC6           ;CTR:  OVERFLOW, ENABLED
4503  024666  052737  100000  002400        BIS     #100000,$FPS    ;SET ERROR BIT
4504  024674  012737  000010  002402        MOV     #10,$FEC        ;SET $FEC
4505  024702  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
4506  024710  010466  000024        30$:    MOV     R4,24(SP)       ;STORE EXP AND SIGN
4507  024714  010166  000026                MOV     R1,26(SP)       ;INSERT LOW ORDER FRACTION
4508  024720  010266  000030                MOV     R2,30(SP)       ;
4509  024724  010366  000032                MOV     R3,32(SP)       ;
4510  024730  000414                        BR      3$              ;
4511  024732  005726                34$:    TST     (SP)+           ;FLUSH SIGN
4512  024734  000402                        BR      29$             ;AND ZERO RESULT
4513  024736  005237  002544        33$:    INC     ADDC7           ;CTR:  UNDERFLOW
4514  024742  005066  000024        29$:    CLR     24(SP)          ;ZERO RESULT
4515  024746  005066  000026                CLR     26(SP)          ;
4516  024752  005066  000030                CLR     30(SP)          ;
4517  024756  005066  000032                CLR     32(SP)          ;
4518  024762  032766  077600  000024  3$:   BIT     #077600,24(SP)  ;SET Z BIT IF EXPONENT ZERO
4519  024770  001003                        BNE     17$             ;
4520  024772  052737  000004  002400        BIS     #04,$FPS        ;
4521  025000  005766  000024        17$:    TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
4522  025004  100003                        BPL     18$             ;
4523  025006  052737  000010  002400        BIS     #10,$FPS        ;
4524  025014  012605                18$:    MOV     (SP)+,R5        ;
4525  025016  012604                        MOV     (SP)+,R4        ;
4526  025020  012603                        MOV     (SP)+,R3        ;RESTORE REGISTERS
4527  025022  012602                        MOV     (SP)+,R2        ;
4528  025024  012601                        MOV     (SP)+,R1        ;
4529  025026  012600                        MOV     (SP)+,R0        ;
4530  025030  062706  000010                ADD     #8.,SP          ;POP SECOND ARGUMENT
4531  025034  105737  002400                TSTB    $FPS            ;FOR D MODE?
4532  025040  100404                        BMI     26$             ;D MODE
4533  025042  012666  000002                MOV     (SP)+,2(SP)     ;F MODE - CONVERT
4534  025046  012666  000002                MOV     (SP)+,2(SP)     ;
4535  025052  000134                26$:    JMP     @(R4)+          ;DONE
4536
4537
4538  025054  005700                14$:    TST     R0              ;CHECK HIGH ORDER FRACTION RESULT
4539  025056  003014                        BGT     22$             ;IF + SIGN OK
4540  025060  001453                        BEQ     23$             ;CHECK FOR ZERO RESULT
4541  025062  005403                        NEG     R3              ;ABS VALUE
4542  025064  005502                        ADC     R2              ;
4543  025066  005501                        ADC     R1              ;
4544  025070  005500                        ADC     R0              ;
4545  025072  005402                        NEG     R2              ;
4546  025074  005501                        ADC     R1              ;
4547  025076  005500                        ADC     R0              ;
```

```
4548  025100  005401                          NEG     R1              ;
4549  025102  005500                          ADC     R0              ;
4550  025104  005400                          NEG     R0              ;
4551  025106  000316                          SWAB    (SP)            ;EXCHANGE SIGNS
4552  025110  030027  000400          22$:    BIT     R0,#400         ;CHECK NORMAL BIT
4553  025114  001427                          BEQ     19$             ;JUMP IF NONE
4554  025116  005704                          TST     R4              ;CHECK FOR UNDERFLOW
4555  025120  003204                          BGT     9$              ;
4556  025122  032737  002000  002400          BIT     #002000,$FPS    ;UNDERFLOW - IS IT ENABLED ?
4557  025130  001702                          BEQ     33$             ;NO, MAKE ZERO RESULT
4558  025132  005237  002544                  INC     ADDC7           ;CTR:  UNDERFLOW
4559  025136  005237  002546                  INC     ADDC8           ;CTR:  UNDERFLOW, ENABLED
4560  025142  052737  100000  002400          BIS     #100000,$FPS    ;SET ERROR BIT
4561  025150  012737  000012  002402          MOV     #12,$FEC        ;SET $FEC
4562  025156  013737  002376  002404          MOV     EXPFEA,$FEA     ;SET $FEA
4563  025164  000304                          SWAB    R4              ;MOVE EXP LEFT
4564  025166  105004                          CLRB    R4              ;CLEAR WHERE FRACTION WILL GO
4565  025170  000137  024610                  JMP     16$             ;ASSEMBLE NUMBER
4566  025174  005304                  19$:    DEC     R4              ;DECREASE EXP
4567  025176  006303                          ASL     R3              ;DOUBLE FRACTION
4568  025200  006102                          ROL     R2              ;
4569  025202  006101                          ROL     R1              ;
4570  025204  006100                          ROL     R0              ;
4571  025206  000740                          BR      22$             ;
4572  025210  162704  000010          23$:    SUB     #8.,R4          ;REDUCE EXP
4573  025214  005701                          TST     R1              ;
4574  025216  001020                          BNE     24$             ;JUMP IF ONLY R0=0
4575  025220  162704  000020                  SUB     #16.,R4         ;
4576  025224  010201                          MOV     R2,R1           ;
4577  025226  001012                          BNE     25$             ;JUMP IF R2 NON ZERO
4578  025230  162704  000020                  SUB     #16.,R4         ;
4579  025234  005703                          TST     R3              ;
4580  025236  001422                          BEQ     21$             ;ANSWER IS ZERO
4581  025240  150301                          BISB    R3,R1           ;MOVE BYTES TO R0,R1
4582  025242  000301                          SWAB    R1              ;
4583  025244  000303                          SWAB    R3              ;
4584  025246  150300                          BISB    R3,R0           ;
4585  025250  005003                          CLR     R3              ;MAKE ALL OTHERS ZERO
4586  025252  000716                          BR      22$             ;GO NORMALIZE
4587  025254  010302                  25$:    MOV     R3,R2           ;
4588  025256  005003                          CLR     R3              ;
4589  025260  000301                  24$:    SWAB    R1              ;MOVE LEFT
4590  025262  150100                          BISB    R1,R0           ;
4591  025264  105001                          CLRB    R1              ;
4592  025266  000302                          SWAB    R2              ;
4593  025270  150201                          BISB    R2,R1           ;
4594  025272  105002                          CLRB    R2              ;
4595  025274  000303                          SWAB    R3              ;
4596  025276  150302                          BISB    R3,R2           ;
4597  025300  105003                          CLRB    R3              ;
4598  025302  000702                          BR      22$             ;
4599  025304  005016                  21$:    CLR     (SP)            ;SET POSITIVE
4600  025306  005004                          CLR     R4              ;
4601  025310  000137  024610                  JMP     16$             ;
4602                                          ;*END OF ADD/SUB
```

```
                        ;*FLOATING POINT SOFTWARE MULTIPLY 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 PRODUCT.
                        ;*
                        ;*  EXAMPLE:     $PUSH
                        ;*               ADDR(OPERAND A)
                        ;*               $PUSH
                        ;*               ADDR(OPERAND B)
                        ;*               $MUL
                        ;*               $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".
                        ;*
                        ;*  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.
```

```
4659                                                    ;*
4660
4661  025314  005237  002550          $MUL:   INC     MULCO           ;CTR:  TOTAL NUMBER MUL
4662  025320  042737  000017  002400          BIC     #17,$FPS        ;CLEAR STATUS BITS N Z V C
4663  025326  105737  002400                  TSTB    $FPS            ;TEST MODE
4664  025332  100402                          BMI     15$             ;D-MODE
4665  025334  004737  027672                  JSR     PC,$CONV        ;F-MODE:  CONVERT 2 F OPDS TO D
4666  025340  010046                  15$:    MOV     R0,-(SP)        ;
4667  025342  010146                          MOV     R1,-(SP)        ;
4668  025344  010246                          MOV     R2,-(SP)        ;SAVE ALL REGISTERS
4669  025346  010346                          MOV     R3,-(SP)        ;
4670  025350  010446                          MOV     R4,-(SP)        ;
4671  025352  010546                          MOV     R5,-(SP)        ;
4672  025354  006366  000014                  ASL     14(SP)          ;SHIFT MULTIPLICAND
4673  025360  006146                          ROL     -(SP)           ;KEEP SIGN
4674  025362  005046                          CLR     -(SP)           ;CLEAR EXPONENT
4675  025364  116616  000021                  MOVB    21(SP),(SP)     ;KEEP MULTIPLICAND EXP
4676  025370  001436                          BEQ     1$              ;ANSWER ZERO
4677  025372  116666  000020  000021          MOVB    20(SP),21(SP)   ;SHIFT FRACTION LEFT
4678  025400  000261                          SEC                     ;INSERT NORMAL BIT
4679  025402  006066  000020                  ROR     20(SP)          ;
4680  025406  116666  000023  000020          MOVB    23(SP),20(SP)   ;
4681  025414  000366  000022                  SWAB    22(SP)          ;
4682  025420  116666  000025  000022          MOVB    25(SP),22(SP)   ;
4683  025426  000366  000024                  SWAB    24(SP)          ;
4684  025432  116666  000027  000024          MOVB    27(SP),24(SP)   ;
4685  025440  000366  000026                  SWAB    26(SP)          ;
4686  025444  105066  000026                  CLRB    26(SP)          ;EXTRA BITS
4687  025450  006366  000030                  ASL     30(SP)          ;SHIFT HIGH MULTIPLIER
4688  025454  005566  000002                  ADC     2(SP)           ;PRODUCT SIGN
4689  025460  105766  000031                  TSTB    31(SP)          ;
4690  025464  001005                          BNE     2$              ;JUMP IF NON ZERO
4691  025466  022626                  1$:     CMP     (SP)+,(SP)+     ;FLUSH SIGN AND EXPONENT
4692  025470  005237  002552                  INC     MULC1           ;CTR:  A=0 AND/OR B=0
4693  025474  000137  026150                  JMP     3$              ;
4694  025500  005000                  2$:     CLR     R0              ;CLEAR PRODUCT
4695  025502  005001                          CLR     R1              ;
4696  025504  005002                          CLR     R2              ;
4697  025506  005003                          CLR     R3              ;
4698  025510  005005                          CLR     R5              ;CLEAR C BIT OVERFLOW CATCHER
4699  025512  006066  000030                  ROR     30(SP)          ;SIGN IS +
4700  025516  012746  000020                  MOV     #16.,-(SP)      ;ITERATION COUNT
4701  025522  016604  000040                  MOV     40(SP),R4       ;GET LOWEST ORDER MULTIPLIER
4702  025526  001404                          BEQ     4$              ;JUMP IF NO BITS HERE
4703  025530  004737  026370                  JSR     PC,30$          ;
4704  025534  012716  000020                  MOV     #16.,(SP)       ;RESTORE COUNT
4705  025540  016604  000036          4$:     MOV     36(SP),R4       ;GET NEXT LOWEST FRACTION
4706  025544  001003                          BNE     5$              ;WORK TO DO
4707  025546  005766  000040                  TST     40(SP)          ;
4708  025552  001406                          BEQ     6$              ;NO PRODUCT YET
4709  025554  004737  026364          5$:     JSR     PC,31$          ;
4710  025560  004737  026272                  JSR     PC,32$          ;ONE BIT FULL PRECISION
4711  025564  012716  000020                  MOV     #16.,(SP)       ;
4712  025570  016604  000034          6$:     MOV     34(SP),R4       ;NEXT TO HIGHEST ORDER FRACTION
4713  025574  001006                          BNE     7$              ;
4714  025576  005766  000036                  TST     36(SP)          ;
```

```
 4715  025602  001003                        BNE     7$                      ;
 4716  025604  005766  000040                TST     40(SP)                  ;
 4717  025610  001402                        BEQ     8$                      ;
 4718  025612  004737  026272          7$:   JSR     PC,32$                  ;
 4719  025616  016604  000032          8$:   MOV     32(SP),R4               ;GET HIGH ORDER BITS
 4720  025622  012716  000007                MOV     #7,(SP)                 ;SEVEN OF THEM
 4721  025626  004737  026272                JSR     PC,32$                  ;
 4722  025632  004737  026276                JSR     PC,33$                  ;NORMAL BIT
 4723  025636  005726                        TST     (SP)+                   ;FLUSH ITERATION COUNT
 4724  025640  062604                        ADD     (SP)+,R4                ;ADD EXP
 4725  025642  006303                        ASL     R3                      ;SHIFT OUT NORMAL BIT
 4726  025644  006102                        ROL     R2                      ;
 4727  025646  006101                        ROL     R1                      ;
 4728  025650  006100                        ROL     R0                      ;
 4729  025652  103405                        BCS     9$                      ;NORMAL BIT FOUND
 4730  025654  006303                        ASL     R3                      ;
 4731  025656  006102                        ROL     R2                      ;
 4732  025660  006101                        ROL     R1                      ;
 4733  025662  006100                        ROL     R0                      ;HAVE IT
 4734  025664  005304                        DEC     R4                      ;ADJUST EXP
 4735  025666  162704  000200          9$:   SUB     #200,R4                 ;REMOVE BIAS FROM EXP
 4736  025672  003022                        BGT     10$                     ;BR IF NO UNDERFLOW
 4737  025674  005237  002560                INC     MULC4                   ;CTR:  UNDERFLOW
 4738  025700  032737  002000  002400        BIT     #002000,$FPS            ;UNDERFLOW - IS IT ENABLED ?
 4739  025706  001517                        BEQ     12$                     ;NO, MAKE ZERO RESULT
 4740  025710  005237  002562                INC     MULC5                   ;CTR:  UNDERFLOW, ENABLED
 4741  025714  052737  100000  002400        BIS     #100000,$FPS            ;SET ERROR BIT
 4742  025722  012737  000012  002402        MOV     #12,$FEC                ;SET $FEC
 4743  025730  013737  002376  002404        MOV     EXPFEA,$FEA             ;SET $FEA
 4744  025736  000427                        BR      11$                     ;CONTINUE
 4745  025740  022704  000377          10$:  CMP     #377,R4                 ;CHECK FOR OVERFLOW
 4746  025744  002024                        BGE     11$                     ;BR IF NO OVERFLOW
 4747  025746  052737  000002  002400        BIS     #02,$FPS                ;SET V BIT ON OVERFLOW
 4748  025754  005237  002554                INC     MULC2                   ;CTR:  OVERFLOW
 4749  025760  032737  001000  002400        BIT     #001000,$FPS            ;OVERFLOW ENABLED ?
 4750  025766  001467                        BEQ     12$                     ;NO, ZERO RESULT
 4751  025770  005237  002556                INC     MULC3                   ;CTR:  OVERFLOW, ENABLED
 4752  025774  052737  100000  002400        BIS     #100000,$FPS            ;SET ERROR BIT
 4753  026002  012737  000010  002402        MOV     #10,$FEC                ;SET $FEC
 4754  026010  013737  002376  002404        MOV     EXPFEA,$FEA             ;SET $FEA
 4755  026016  105003                  11$:  CLRB    R3                      ;
 4756  026020  150203                        BISB    R2,R3                   ;SHIFT FRACTION RIGHT
 4757  026022  000303                        SWAB    R3                      ;
 4758  026024  105002                        CLRB    R2                      ;
 4759  026026  150102                        BISB    R1,R2                   ;
 4760  026030  000302                        SWAB    R2                      ;
 4761  026032  105001                        CLRB    R1                      ;
 4762  026034  150001                        BISB    R0,R1                   ;
 4763  026036  000301                        SWAB    R1                      ;
 4764  026040  105000                        CLRB    R0                      ;
 4765  026042  150400                        BISB    R4,R0                   ;
 4766  026044  000300                        SWAB    R0                      ;
 4767  026046  006026                        ROR     (SP)+                   ;GET PRODUCT SIGN
 4768  026050  006000                        ROR     R0                      ;PUT IN RESULT
 4769  026052  006001                        ROR     R1                      ;
 4770  026054  006002                        ROR     R2                      ;
```

```
4771  026056  006003                          ROR     R3
4772  026060  005503                          ADC     R3              ;ROUND RESULT
4773  026062  005502                          ADC     R2
4774  026064  005501                          ADC     R1
4775  026066  005500                          ADC     R0
4776  026070  103401                          BCS     16$             ;OVERFLOW ON ROUND ?
4777  026072  102032                          BVC     13$
4778  026074  052737  000002  002400  16$:    BIS     #02,$FPS        ;YES - SET V BIT
4779  026102  005237  002554                  INC     MULC2           ;CTR:  OVERFLOW
4780  026106  032737  001000  002400          BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
4781  026114  001415                          BEQ     3$              ;NO, ZERO RESULT
4782  026116  005237  002556                  INC     MULC3           ;CTR:  OVERFLOW, ENABLED
4783  026122  052737  100000  002400          BIS     #100000,$FPS    ;SET ERROR BIT
4784  026130  012737  000010  002402          MOV     #10,$FEC        ;SET $FEC
4785  026136  013737  002376  002404          MOV     EXPFEA,$FEA     ;SET $FEA
4786  026144  000405                          BR      13$             ;CONTINUE
4787  026146  005726                  12$:    TST     (SP)+           ;FLUSH SIGN
4788  026150  005000                  3$:     CLR     R0              ;CLEAR RESUL
4789  026152  005001                          CLR     R1
4790  026154  005002                          CLR     R2
4791  026156  005003                          CLR     R3
4792  026160  010066  000024          13$:    MOV     R0,24(SP)
4793  026164  010166  000026                  MOV     R1,26(SP)       ;STUFF RESULT
4794  026170  010266  000030                  MOV     R2,30(SP)
4795  026174  010366  000032                  MOV     R3,32(SP)
4796  026200  032766  077600  000024          BIT     #077600,24(SP)  ;SET Z BIT IF EXPONENT ZERO
4797  026206  001003                          BNE     17$
4798  026210  052737  000004  002400          BIS     #04,$FPS
4799  026216  005766  000024          17$:    TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
4800  026222  100003                          BPL     18$
4801  026224  052737  000010  002400          BIS     #10,$FPS
4802  026232  012605                  18$:    MOV     (SP)+,R5
4803  026234  012604                          MOV     (SP)+,R4
4804  026236  012603                          MOV     (SP)+,R3        ;RESTORE REGISTERS
4805  026240  012602                          MOV     (SP)+,R2
4806  026242  012601                          MOV     (SP)+,R1
4807  026244  012600                          MOV     (SP)+,R0
4808  026246  062706  000010                  ADD     #8.,SP          ;CLEAR SECOND OPERAND OFF STACK
4809  026252  105737  002400                  TSTB    $FPS            ;F OR D MODE
4810  026256  100404                          BMI     14$             ;D-MODE
4811  026260  012666  000002                  MOV     (SP)+,2(SP)     ;F MODE - CONVERT
4812  026264  012666  000002                  MOV     (SP)+,2(SP)
4813  026270  000134                  14$:    JMP     @(R4)+          ;RETURN
4814
4815  026272  006204                  32$:    ASR     R4              ;TEST NEXT MULTIPLIER BIT
4816  026274  103022                          BCC     34$             ;JUMP IF ZERO
4817  026276  066603  000032          33$:    ADD     32(SP),R3       ;ADD IN MULTIPLICAND
4818  026302  005502                          ADC     R2
4819  026304  005501                          ADC     R1
4820  026306  005500                          ADC     R0
4821  026310  005505                          ADC     R5              ;SAVE OVERFLOW
4822  026312  066602  000030                  ADD     30(SP),R2
4823  026316  005501                          ADC     R1
4824  026320  005500                          ADC     R0
4825  026322  005505                          ADC     R5
4826  026324  066601  000026                  ADD     26(SP),R1
```

FLOATING POINT SOFTWARE ROUTINES

```
4827  026330  005500                        ADC     R0              ;
4828  026332  005505                        ADC     R5              ;
4829  026334  066600  000024                ADD     24(SP),R0       ;
4830  026340  005505                        ADC     R5              ;
4831  026342  006205              34$:      ASR     R5              ;RECOVER OVERFLOW IF ANY
4832  026344  006000                        ROR     R0              ;SHIFT PRODUCT
4833  026346  006001                        ROR     R1              ;
4834  026350  006002                        ROR     R2              ;
4835  026352  006003                        ROR     R3              ;
4836  026354  005366  000002                DEC     2(SP)           ;COUNT LOOP
4837  026360  003344                        BGT     32$             ;AGAIN
4838  026362  000207                        RTS     PC              ;RETURN
4839  026364  005366  000002      31$:      DEC     2(SP)           ;ONLY 15 BITS THIS PASS
4840  026370  006204              30$:      ASR     R4              ;TEST NEXT MULTIPLIER BIT
4841  026372  103007                        BCC     35$             ;JUMP IF ZERO
4842  026374  066601  000026                ADD     26(SP),R1       ;USE ONLY HIGH ORDER MULTIPLICAND
4843  026400  005500                        ADC     R0              ;
4844  026402  005505                        ADC     R5              ;
4845  026404  066600  000024                ADD     24(SP),R0       ;
4846  026410  005505                        ADC     R5              ;
4847  026412  006205              35$:      ASR     R5              ;RECOVER ANY OVERFLOW
4848  026414  006000                        ROR     R0              ;
4849  026416  006001                        ROR     R1              ;
4850  026420  006002                        ROR     R2              ;
4851  026422  006003                        ROR     R3              ;
4852  026424  005366  000002                DEC     2(SP)           ;COUNT LOOP
4853  026430  003357                        BGT     30$             ;
4854  026432  000207                        RTS     PC              ;RETURN
4855                                        ;*END OF MUL
```

```
4856                    ;*FLOATING POINT SOFTWARE DIVIDE ROUTINE
4857                    ;*
4858                    ;*  THIS ROUTINE TAKES THE TOP TWO ELEMENTS ON THE STACK
4859                    ;*    (LENGTH OF 2 OR 4 WORDS DEPENDING UPON WHETHER THE
4860                    ;*    F/D MODE BIT7 IN $FPS IS 0 OR 1 RESPECTIVELY) AND
4861                    ;*    REPLACES THEM WITH THEIR QUOTIENT.
4862                    ;*
4863                    ;*  EXAMPLE:    $PUSH
4864                    ;*              ADDR(OPERAND A)
4865                    ;*              $PUSH
4866                    ;*              ADDR(OPERAND B)
4867                    ;*              $DIV
4868                    ;*              $POPX
4869                    ;*              ADDR(RESULT)
4870                    ;*                RESULT=A/B
4871                    ;*
4872                    ;*  NOTE ROUTINE IS CALLED THROUGH POLISH MODE OPERATORS.
4873                    ;*
4874                    ;*  ALL OPERATIONS ARE CARRIED OUT IN DOUBLE (LENGTH=4)
4875                    ;*    PRECISION MODE.  SINGLE PRECISION OPERANDS ARE
4876                    ;*    EXTENDED (VIA $CONV SUBROUTINE) WITH LOW-ORDER
4877                    ;*    ZEROS, AND LATER TRUNCATED BACK TO SINGLE PRECISION
4878                    ;*    FROM THE DOUBLE PRECISION RESULT.
4879                    ;*
4880                    ;*  STATUS BITS:
4881                    ;*    THE N, Z, V, AND C BITS OF $FPS ARE SET AS FOLLOWS:
4882                    ;*    N = 1  IF RESULT NEGATIVE (IE, BIT15 = 1),
4883                    ;*           ELSE  N = 0
4884                    ;*    Z = 1  IF RESULT ZERO (IE, EXPONENT = 000(8)),
4885                    ;*           ELSE  Z = 0
4886                    ;*    V = 1  IF ARITHMETIC OVERFLOW OCCURRED,
4887                    ;*           ELSE  V = 0
4888                    ;*    C = 0  ALWAYS
4889                    ;*
4890                    ;*  ERROR CONDITIONS:
4891                    ;*    IF AN ARITHMETIC OVERFLOW CONDITION OCCURS, THE RESULT
4892                    ;*    WILL BE SET TO ZERO IF THE OVERFLOW ENABLE BIT OF $FPS
4893                    ;*    (BIT09) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
4894                    ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE UNDERBIASED
4895                    ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
4896                    ;*    $FEC WILL BE SET TO 10(8), AND $FEA WILL BE LOADED WITH
4897                    ;*    THE VALUE IN LOCATION "EXPFEA".  IN EITHER INSTANCE,
4898                    ;*    THE V-BIT (BIT01) WILL BE SET.
4899                    ;*    IF AN ARITHMETIC UNDERFLOW CONDITION OCCURS, THE RESULT
4900                    ;*    WILL BE SET TO ZERO IF THE UNDERFLOW ENABLE BIT OF $FPS
4901                    ;*    (BIT10) IS ZERO.  IF THIS BIT IS SET, THEN THE RESULT
4902                    ;*    WILL BE CORRECT, EXCEPT THE EXPONENT WILL BE OVERBIASED
4903                    ;*    BY 400(8).  ALSO, THE $FPS ERROR BIT (BIT15) WILL BE SET,
4904                    ;*    $FEC WILL BE SET TO 12(8), AND $FEA WILL BE LOADED WITH
4905                    ;*    THE VALUE IN LOCATION "EXPFEA".
4906                    ;*    IF DIVISION BY ZERO IS ATTEMPTED (EG, EXPONENT OF
4907                    ;*    DENOMINATOR OPERAND IS ZERO), THE RESULT LEFT ON THE
4908                    ;*    STACK WILL BE THE NUMERATOR, WITH THE CONDITION CODES SET
4909                    ;*    ACCORDINGLY.  THE $FPS ERROR BIT WILL BE SET, $FEC WILL BE
4910                    ;*    SET TO 4(8), AND $FEA WILL BE SET TO THE VALUE CONTAINED
4911                    ;*    IN THE LOCATION "EXPFEA".
```

```
4912                                                    ;*
4913                                                    ;*  ALL REGISTERS ARE PRESERVED UPON ENTRY, AND RESTORED UPON
4914                                                    ;*    EXIT.  THE ROUTINES ARE RE-ENTRANT.
4915                                                    ;*
4916                                                    ;*  ADAPTED FROM PDP-11 FORTRAN SOFTWARE
4917                                                    ;*    BY DONALD NORTH, SEPTEMBER, 1975.
4918                                                    ;*
4919
4920  026434  005237  002564          $DIV:   INC     DIVC0           ;CTR:  TOTAL NUMBER OF DIV
4921  026440  042737  000017  002400        BIC     #17,$FPS        ;CLEAR STATJS BITS N Z V C
4922  026446  105737  002400                TSTB    $FPS            ;TEST MODE
4923  026452  100402                        BMI     14$             ;D-MODE
4924  026454  004737  027672                JSR     PC,$CONV        ;F-MODE:  CONVERT 2 F OPDS TO D
4925  026460  010046          14$:    MOV     R0,-(SP)        ;
4926  026462  010146                  MOV     R1,-(SP)        ;
4927  026464  010246                  MOV     R2,-(SP)        ;
4928  026466  010346                  MOV     R3,-(SP)        ;SAVE REGISTERS
4929  026470  010446                  MOV     R4,-(SP)        ;
4930  026472  010546                  MOV     R5,-(SP)        ;
4931  026474  032766  077600  000014        BIT     #077600,14(SP)  ;DIVIDE BY ZERO ?
4932  026502  001015                        BNE     2$              ;NO
4933  026504  005237  002570                INC     DIVC2           ;CTR:  DENOM=0
4934  026510  052737  100000  002400        BIS     #100000,$FPS    ;YES, SET ERROR BIT
4935  026516  012737  000004  002402        MOV     #4,$FEC         ;SET $FEC
4936  026524  013737  002376  002404        MOV     EXPFEA,$FEA     ;SET $FEA
4937  026532  000137  027442                JMP     9$              ;DONE
4938  026536  005000          2$:     CLR     R0              ;
4939  026540  005001                  CLR     R1              ;
4940  026542  005002                  CLR     R2              ;
4941  026544  005003                  CLR     R3              ;
4942  026546  005046                  CLR     -(SP)           ;
4943  026550  006366  00002E                ASL     26(SP)          ;SHIFT NUMERATOR
4944  026554  006116                        ROL     (SP)            ;NUMERATOR SIGN
4945  026556  005046                        CLR     -(SP)           ;
4946  026560  156616  000031                BISB    31(SP),(SP)     ;NUMERATOR EXP
4947  026564  001004                        BNE     6$              ;NUMERATOR IS ZERO?
4948  026566  005237  002566                INC     DIVC1           ;CTR:  NUMER=0
4949  026572  000137  027416                JMP     1$              ;YES, DONE
4950  026576  156600  000030          6$:     BISB    30(SP),R0       ;
4951  026602  000300                        SWAB    R0              ;LEFT JUSTIFY NUMERATOR FRACTION
4952  026604  000261                        SEC                     ;INSERT NORMAL BIT
4953  026606  006000                        ROR     R0              ;
4954  026610  156600  000033                BISB    33(SP),R0       ;
4955  026614  156601  000032                BISB    32(SP),R1       ;
4956  026620  000301                        SWAB    R1              ;
4957  026622  156601  000035                BISB    35(SP),R1       ;
4958  026626  156602  000034                BISB    34(SP),R2       ;
4959  026632  000302                        SWAB    R2              ;
4960  026634  156602  000037                BISB    37(SP),R2       ;
4961  026640  156603  000036                BISB    36(SP),R3       ;
4962  026644  000303                        SWAB    R3              ;
4963  026646  006366  000020                ASL     20(SP)          ;SHIFT DENOMINATOR
4964  026652  005566  000002                ADC     2(SP)           ;RESULT SIGN
4965  026656  005004                        CLR     R4              ;
4966  026660  156604  000021                BISB    21(SP),R4       ;DIVISOR EXPONENT
4967  026664  160416                        SUB     R4,(SP)         ;SUBTRACT EXPONENTS
```

```
 4968  026666  000366  000020                  SWAB    20(SP)          ;LEFT JUSTIFY DENOM
 4969  026672  000261                          SEC                     ;INSERT NORMAL BIT
 4970  026674  006066  000020                  ROR     20(SP)          ;
 4971  026700  116666  000023  000020          MOVB    23(SP),20(SP)   ;
 4972  026706  116666  000022  000023          MOVB    22(SP),23(SP)   ;
 4973  026714  116666  000025  000022          MOVB    25(SP),22(SP)   ;
 4974  026722  116666  000024  000025          MOVB    24(SP),25(SP)   ;
 4975  026730  116666  000027  000024          MOVB    27(SP),24(SP)   ;
 4976  026736  116666  000026  000027          MOVB    26(SP),27(SP)   ;
 4977  026744  105066  000026                  CLRB    26(SP)          ;
 4978  026750  005066  000030                  CLR     30(SP)          ;CLEAR QUOTIENT
 4979  026754  005066  000032                  CLR     32(SP)          ;
 4980  026760  005066  000034                  CLR     34(SP)          ;
 4981  026764  020066  000020                  CMP     R0,20(SP)       ;COMPARE HIGH NUM + DENOM
 4982  026770  101020                          BHI     3$              ;JUMP IF DENOM LOW
 4983  026772  103424                          BLO     4$              ;JUMP IF DENOM HI
 4984  026774  020166  000022                  CMP     R1,22(SP)       ;COMPARE LOW ORDER PARTS
 4985  027000  101014                          BHI     3$              ;
 4986  027002  103420                          BLO     4$              ;
 4987  027004  020266  000024                  CMP     R2,24(SP)       ;
 4988  027010  101010                          BHI     3$              ;
 4989  027012  103414                          BLO     4$              ;
 4990  027014  020366  000026                  CMP     R3,26(SP)       ;
 4991  027020  101004                          BHI     3$              ;
 4992  027022  001010                          BNE     4$              ;
 4993  027024  005216                          INC     (SP)            ;BUMP EXP
 4994  027026  005004                          CLR     R4              ;
 4995  027030  000443                          BR      5$              ;
 4996
 4997  027032  006000                  3$:     ROR     R0              ;HALVE DENOM (C=0)
 4998  027034  006001                          ROR     R1              ;TO INSURE N<D
 4999  027036  006002                          ROR     R2              ;
 5000  027040  006003                          ROR     R3              ;
 5001  027042  005216                          INC     (SP)            ;COMPENSATE EXP
 5002  027044  012705  000011          4$:     MOV     #9.,R5          ;FIRST NINE QUOTIENT BITS
 5003  027050  004737  027534                  JSR     PC,30$          ;
 5004  027054  110466  000030                  MOVB    R4,30(SP)       ;SAVE ALL HIGH ORDER Q FRACTION
 5005  027060  005705                          TST     R5              ;DONE?
 5006  027062  001025                          BNE     10$             ;YES - REST OF NUMBER IS 0
 5007  027064  012705  000020                  MOV     #16.,R5         ;16 MORE BITS
 5008  027070  004737  027534                  JSR     PC,30$          ;
 5009  027074  010466  000032                  MOV     R4,32(SP)       ;
 5010  027100  005705                          TST     R5              ;
 5011  027102  001015                          BNE     10$             ;
 5012  027104  012705  000020                  MOV     #16.,R5         ;
 5013  027110  004737  027534                  JSR     PC,30$          ;
 5014  027114  010466  000034                  MOV     R4,34(SP)       ;
 5015  027120  005705                          TST     R5              ;
 5016  027122  001005                          BNE     10$             ;
 5017  027124  012705  000020                  MOV     #16.,R5         ;
 5018  027130  004737  027534                  JSR     PC,30$          ;
 5019  027134  000401                          BR      5$              ;
 5020  027136  005004                  10$:    CLR     R4              ;CLEAR LOWEST ORDER QUOTIENT
 5021  027140  012605                  5$:     MOV     (SP)+,R5        ;PUSH UP EXPONENT
 5022  027142  062705  000200                  ADD     #200,R5         ;INSERT BIAS
 5023  027146  003022                          BGT     7$              ;BR IF NO UNDERFLOW
```

```
5024  027150  005237  002576                    INC     DIVC5           ;CTR:  UNDERFLOW
5025  027154  032737  002000  002400            BIT     #002000,$FPS    ;UNDERFLOW - IS IT ENABLED ?
5026  027162  001516                            BEQ     15$             ;NO, MAKE ZERO RESULT
5027  027164  005237  002600                    INC     DIVC6           ;CTR:  UNDERFLOW, ENABLED
5028  027170  052737  100000  002400            BIS     #100000,$FPS    ;SET ERROR BIT
5029  027176  012737  000012  002402            MOV     #12,$FEC        ;SET $FEC
5030  027204  013737  002376  002404            MOV     EXPFEA,$FEA     ;SET $FEA
5031  027212  000427                            BR      11$             ;CONTINUE
5032  027214  022705  000377            7$:     CMP     #377,R5         ;CHECK FOR OVERFLOW
5033  027220  002024                            BGE     11$             ;BR IF NO OVERFLOW
5034  027222  052737  000002  002400            BIS     #02,$FPS        ;SET V BIT ON OVERFLOW
5035  027230  005237  002572                    INC     DIVC3           ;CTR:  OVERFLOW
5036  027234  032737  001000  002400            BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
5037  027242  001466                            BEQ     15$             ;NO, ZERO RESULT
5038  027244  005237  002574                    INC     DIVC4           ;CTR:  OVERFLOW, ENABLED
5039  027250  052737  100000  002400            BIS     #100000,$FPS    ;SET ERROR BIT
5040  027256  012737  000010  002402            MOV     #10,$FEC        ;SET $FEC
5041  027264  013737  002376  002404            MOV     EXPFEA,$FEA     ;SET $FEA
5042  027272  110566  000027            11$:    MOVB    R5,27(SP)       ;PUT EXPIN RESULT
5043  027276  006026                            ROR     (SP)+           ;INSERT SIGN
5044  027300  006066  000024                    ROR     24(SP)          ;
5045  027304  006066  000026                    ROR     26(SP)          ;
5046  027310  006066  000030                    ROR     30(SP)          ;
5047  027314  006004                            ROR     R4              ;
5048  027316  005504                            ADC     R4              ;ROUND
5049  027320  005566  000030                    ADC     30(SP)          ;
5050  027324  005566  000026                    ADC     26(SP)          ;
5051  027330  005566  000024                    ADC     24(SP)          ;
5052  027334  010466  000032                    MOV     R4,32(SP)       ;INSERT LOW ORDER FRACTION
5053  027340  103401                            BCS     16$             ;OVERFLOW ON ROUND ?
5054  027342  102037                            BVC     9$              ;
5055  027344  052737  000002  002400    16$:    BIS     #02,$FPS        ;YES - SET V BIT
5056  027352  005237  002572                    INC     DIVC3           ;CTR:  OVERFLOW
5057  027356  032737  001000  002400            BIT     #001000,$FPS    ;OVERFLOW ENABLED ?
5058  027364  001416                            BEQ     19$             ;NO, ZERO RESULT
5059  027366  005237  002574                    INC     DIVC4           ;CTR:  OVERFLOW, ENABLED
5060  027372  052737  100000  002400            BIS     #100000,$FPS    ;SET ERROR BIT
5061  027400  012737  000010  002402            MOV     #10,$FEC        ;SET $FEC
5062  027406  013737  002376  002404            MOV     EXPFEA,$FEA     ;SET $FEA
5063  027414  000412                            BR      9$              ;CONTINUE
5064  027416  005726                    1$:     TST     (SP)+           ;FLUSH EXP
5065  027420  005726                    15$:    TST     (SP)+           ;FLUSH SIGN
5066  027422  005066  000024            19$:    CLR     24(SP)          ;CLEAR RESULT
5067  027426  005066  000026                    CLR     26(SP)          ;
5068  027432  005066  000030                    CLR     30(SP)          ;
5069  027436  005066  000032                    CLR     32(SP)          ;
5070  027442  032766  077600  000024    9$:     BIT     #077600,24(SP)  ;SET Z BIT IF EXPONENT ZERO
5071  027450  001003                            BNE     17$             ;
5072  027452  052737  000004  002400            BIS     #04,$FPS        ;
5073  027460  005766  000024            17$:    TST     24(SP)          ;SET N BIT IF RESULT NEGATIVE
5074  027464  100003                            BPL     18$             ;
5075  027466  052737  000010  002400            BIS     #10,$FPS        ;
5076  027474  012605                    18$:    MOV     (SP)+,R5        ;
5077  027476  012604                            MOV     (SP)+,R4        ;
5078  027500  012603                            MOV     (SP)+,R3        ;RESTORE REGISTERS
5079  027502  012602                            MOV     (SP)+,R2        ;
```

```
5080  027504  012601                        MOV     (SP)+,R1        ;
5081  027506  012600                        MOV     (SP)+,R0        ;
5082  027510  062706  000010                ADD     #8.,SP          ;FLUSH FIRST ARG
5083  027514  105737  002400                TSTB    $FPS            ;F OR D MODE
5084  027520  100404                        BMI     13$             ;D MODE
5085  027522  012666  000002                MOV     (SP)+,2(SP)     ;F MODE - CONVERT
5086  027526  012666  000002                MOV     (SP)+,2(SP)     ;
5087  027532  000134              13$:      JMP     @(R4)+          ;RETURN
5088
5089  027534  006304              30$:      ASL     R4              ;SHIFT QUOTIENT
5090  027536  006303                        ASL     R3              ;
5091  027540  006102                        ROL     R2              ;
5092  027542  006101                        ROL     R1              ;
5093  027544  006100                        ROL     R0              ;
5094  027546  103420                        BCS     31$             ;GUARANTEED TO GO
5095  027550  026600  000022                CMP     22(SP),R0       ;COMPARE HIGH DIVISOR AND DIVIDEND
5096  027554  101034                        BHI     32$             ;DIVISOR BIGGER
5097  027556  103414                        BLO     31$             ;
5098  027560  026601  000024                CMP     24(SP),R1       ;CHECK LOW ORDERS
5099  027564  101030                        BHI     32$             ;
5100  027566  103410                        BLO     31$             ;
5101  027570  026602  000026                CMP     26(SP),R2       ;
5102  027574  101024                        BHI     32$             ;
5103  027576  103404                        BLO     31$             ;
5104  027600  026603  000030                CMP     30(SP),R3       ;
5105  027604  101020                        BHI     32$             ;
5106  027606  001422                        BEQ     33$             ;NUMER=DENOM
5107  027610  166603  000030      31$:      SUB     30(SP),R3       ;N=N-D
5108  027614  005602                        SBC     R2              ;
5109  027616  005601                        SBC     R1              ;
5110  027620  005600                        SBC     R0              ;
5111  027622  166602  000026                SUB     26(SP),R2       ;
5112  027626  005601                        SBC     R1              ;
5113  027630  005600                        SBC     R0              ;
5114  027632  166601  000024                SUB     24(SP),R1       ;
5115  027636  005600                        SBC     R0              ;
5116  027640  166600  000022                SUB     22(SP),R0       ;
5117  027644  005204                        INC     R4              ;INSERT QUOTIENT BIT
5118  027646  005305              32$:      DEC     R5              ;COUNT LOOP
5119  027650  003331                        BGT     30$             ;
5120  027652  000207                        RTS     PC              ;RETURN
5121  027654  005204              33$:      INC     R4              ;INSERT LAST 1 BIT IN QUOTIENT
5122  027656  000401                        BR      34$             ;
5123  027660  006304              35$:      ASL     R4              ;FINISH OUT QUOTIENT WITH ZEROS
5124  027662  005305              34$:      DEC     R5              ;
5125  027664  003375                        BGT     35$             ;
5126  027666  005205                        INC     R5              ;FLAG NO MORE NUMER
5127  027670  000207                        RTS     PC              ;RETURN
5128                                        ;*END OF DIV
```

```
                                        ;*CONVERT TOP 2 OPERANDS ON STACK FROM F MODE
                                        ;*    TO D MODE
                                        ;*
                                        ;*  THIS ROUTINE TAKES THE TOP TWO 2-WORD (SINGLE
                                        ;*    PRECISION) FLOATING POINT NUMBERS ON THE
                                        ;*    STACK AND CONVERTS THEM BOTH TO 4-WORD
                                        ;*    (DOUBLE PRECISION) FORMAT BY APPENDING
                                        ;*    TWO WORDS OF ZEROS AS THE LOW ORDER BIT
                                        ;*    EXTENSION.
                                        ;*

027672  005046                 $CONV:  CLR     -(SP)           ;CLEAR WORD 3 OF A
027674  016646  000006                 MOV     6(SP),-(SP)     ;MOVE WORD 2 OF A
027700  016646  000006                 MOV     6(SP),-(SP)     ;MOVE WORD 1 OF A
027704  016646  000006                 MOV     6(SP),-(SP)     ;MOVE RETURN ADDR TO TOP
027710  016666  000020  000014         MOV     20(SP),14(SP)   ;MOVE WORD 2 OF B
027716  016666  000016  000012         MOV     16(SP),12(SP)   ;MOVE WORD 1 OF B
027724  005066  000020                 CLR     20(SP)          ;CLEAR WORD 4 OF B
027730  005066  000016                 CLR     16(SP)          ;CLEAR WORD 3 OF B
027734  005066  000010                 CLR     10(SP)          ;CLEAR WORD 4 OF A
027740  000207                         RTS     PC              ;
```

```
                                        SCOPE HANDLER ROUTINE

  5150                                  .SBTTL  SCOPE HANDLER ROUTINE
  5151
  5152                                  ;;**********************************************************************
  5153                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  5154                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<15:0>)
  5155                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  5156                                  ;*SW14=1          LOOP ON TEST
  5157                                  ;*SW11=1          INHIBIT ITERATIONS
  5158                                  ;*SW09=1          LOOP ON ERROR
  5159                                  ;*SW08=1          LOOP ON TEST IN "$LPTST"
  5160                                  ;*CALL
  5161                                  ;*      SCOPE               ;;SCOPE=IOT
  5162
  5163  027742                          $SCOPE:
  5164  027742                          64$:
  5165  027742  032777  040000  151176  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
  5166  027750  001114                          BNE     $OVER           ;;YES IF SW14=1
  5167                                  ;*****START OF CODE FOR THE XOR TESTER*****
  5168  027752  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
  5169                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
  5170  027754  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  5171  027760  012737  030000  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
  5172  027766  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
  5173  027772  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  5174  027776  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
  5175  030000  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
  5176  030002  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  5177  030006  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
  5178  030010                          6$:;*****END OF CODE FOR THE XOR TESTER*****
  5179  030010  032777  000400  151130          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
  5180  030016  001404                          BEQ     2$              ;;BR IF NO
  5181  030020  023737  001112  001102          CMP     $LPTST,$TSTNM   ;;ON THE RIGHT TEST?
  5182  030026  001465                          BEQ     $OVER           ;;BR IF YES
  5183  030030  005737  001104          2$:     TST     $ERFLG          ;;HAS AN ERROR OCCURRED?
  5184  030034  001421                          BEQ     3$              ;;BR IF NO
  5185  030036  023737  001122  001104          CMP     $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  5186  030044  101015                          BHI     3$              ;;BR IF NO
  5187  030046  032777  001000  151072          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
  5188  030054  001404                          BEQ     4$              ;;BR IF NO
  5189  030056  013737  001114  001110  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
  5190  030064  000446                          BR      $OVER
  5191  030066  005037  001104          4$:     CLR     $ERFLG          ;;ZERO THE ERROR FLAG
  5192  030072  005037  001166                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  5193  030076  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
  5194  030100  032777  004000  151040  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
  5195  030106  001011                          BNE     1$              ;;BR IF YES
  5196  030110  005737  001210                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
  5197  030114  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
  5198  030116  005237  001106                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
  5199  030122  023737  001166  001106          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
  5200  030130  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
  5201  030132  012737  000001  001106  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
  5202  030140  013737  030216  001166          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
  5203  030146  005237  001102          $SVLAD: INC     $TSTNM          ;;COUNT TEST NUMBERS
  5204  030152  013737  001102  001206          MOV     $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
  5205  030160  011637  001110                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
 5206  030164  011637  001114                        MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 5207  030170  005037  001170                        CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 5208  030174  012737  000001  001122                MOV     #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 5209  030202  013777  001102  150740  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 5210  030210  013716  001110                        MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 5211  030214  000002                                RTI                     ;;FIXES PS
 5212  030216  000400                        $MXCNT: 400                     ;;MAX. NUMBER OF ITERATIONS
```

```
                                        ERROR HANDLER ROUTINE

                                        .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

030220                                  $ERROR:
030220  010037  002602                          MOV     R0,EREG0        ; DISPLAY R0
030224  010137  002604                          MOV     R1,EREG1        ;         R1
030230  010237  002606                          MOV     R2,EREG2        ;         R2
030234  010337  002610                          MOV     R3,EREG3        ;         R3
030240  010437  002612                          MOV     R4,EREG4        ;         R4
030244  010537  002614                          MOV     R5,EREG5        ;         R5
030250  010637  002616                          MOV     R6,EREG6        ; GET R6(SP) BEFORE TRAP
030254  062737  000004  002616                  ADD     #4,EREG6        ;
030262  011637  002620                          MOV     (SP),EREG7      ; PC -> ERROR CALL INSTR
030266  005237  001104                  7$:     INC     $ERFLG          ;;SET THE ERROR FLAG
030272  001775                                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
030274  013777  001102  150646                  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
030302  032777  002000  150636                  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
030310  001402                                  BEQ     1$              ;;NO - SKIP
030312  104401  001172                          TYPE    ,$BELL          ;;RING BELL
030316  005237  001116                  1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
030322  011637  001124                          MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
030326  162737  000002  001124                  SUB     #2,$ERRPC
030334  117737  150564  001120                  MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
030342  032777  020000  150576                  BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
030350  001004                                  BNE     20$             ;;SKIP TYPEOUTS
030352  004737  030462                          JSR     PC,$TYPERR      ;;GO TO USER ERROR ROUTINE
030356  104401  001177                          TYPE    ,$CRLF
030362                                  20$:
030362  122737  000001  001222                  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
030370  001007                                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
030372  113737  001120  030404                  MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
030400  004737  031222                          JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
030404     000                          21$:    .BYTE   0
030405     000                                  .BYTE   0
030406  000777                          22$:    BR      22$             ;;APT ERROR LOOP
030410  005777  150532                  2$:     TST     @SWR            ;;HALT ON ERROR
030414  100001                                  BPL     3$              ;;SKIP IF CONTINUE
030416  000000                                  HALT                    ;;HALT ON ERROR!
030420  032777  001000  150520          3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
030426  001402                                  BEQ     4$              ;;BR IF NO
030430  013716  001114                          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
030434  005737  001170                  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
030440  001402                                  BEQ     5$              ;;BR IF NONE
030442  013716  001170                          MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
030446                                  5$:
```

```
 5269  030446  022737  022160  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 5270  030454  001001                          BNE     6$              ;;BRANCH IF NO
 5271  030456  000000                          HALT                    ;;YES
 5272  030460                          6$:
 5273  030460  000002                  64$:    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 LAST

030462                          $TYPERR:
030462  104401                          TYPE                    ;START WITH MESSAGE PREFIX, HOT OR WARM
030464  001177          HOTWRM: .WORD   $CRLF                   ;PTR TO "HOT" OR "WARM"
030466  010046                  MOV     R0,-(SP)                ;SAVE R0
030470  010146                  MOV     R1,-(SP)                ;SAVE R1
030472  005000                  CLR     R0                      ;PICKUP ITEM INDEX
030474  153700  001120          BISB    @#$ITEMB,R0             ;
030500  001004                  BNE     1$                      ;IF ITEM NUMBER FROM ERROR 0,
                                                                ;  JUST TYPE PC OF ERROR
030502  013746  001124          MOV     $ERRPC,-(SP)            ;GET ERROR PC FOR TYPEOUT
030506  104402                  TYPOC                           ;TYPE OCTAL, ALL DIGITS
030510  000454                  BR      7$                      ;EXIT
030512  005300          1$:     DEC     R0                      ;ADJUST ERROR # FOR TABLE INDEX
030514  006300                  ASL     R0                      ;  OF 20. BYTES/ENTRY
030516  006300                  ASL     R0                      ;
030520  010001                  MOV     R0,R1                   ;
030522  006300                  ASL     R0                      ;
030524  006300                  ASL     R0                      ;
030526  060100                  ADD     R1,R0                   ;
030530  062700  001232          ADD     #$ERRTB,R0              ;FORM TABLE PTR
030534  012037  030544          MOV     (R0)+,2$                ;PICKUP "ERROR MESSAGE" PTR
030540  001404                  BEQ     3$                      ;SKIP TYPEOUT IF NULL
030542  104401                  TYPE                            ;TYPE "ERROR MESSAGE"
030544  000000          2$:     .WORD   0                       ;"ERROR MESSAGE" PTR HERE
030546  104401  001177          TYPE    ,$CRLF                  ;CR & LF
030552  104401  030702  3$:     TYPE    ,11$                    ;"TEST #    ERR PC" HEADER
030556  012037  030566          MOV     (R0)+,4$                ;PICKUP "DATA HEADER" PTR
030562  001402                  BEQ     5$                      ;SKIP TYPEOUT IF NULL
030564  104401                  TYPE                            ;TYPE "DATA HEADER"
030566  000000          4$:     .WORD   0                       ;"DATA HEADER" PTR HERE
030570  104401  001177  5$:     TYPE    $CRLF                   ;CR & LF
030574  017746  000074          MOV     @3$,-(SP)               ;($TESTN)
```

ERROR MESSAGE TYPEOUT ROUTINE (MODIFIED SYSMAC)

```
5330  030600  104402                          TYPOC                   ;OCTAL W/ LEADING ZEROS
5331  030602  104401  030700                  TYPE    ,10$            ;<HT>
5332  030606  017746  000064                  MOV     @9$,-(SP)       ;($ERRPC)
5333  030612  104402                          TYPOC                   ;OCTAL W/ LEADING ZEROS
5334  030614  104401  030700                  TYPE    ,10$            ;<HT>
5335  030620  012001                          MOV     (R0)+,R1        ;PICKUP "DATA TABLE" PTR
5336  030622  001407                          BEQ     7$              ;EXIT IF NULL
5337  030624  013146                  6$:     MOV     @(R1)+ -(SP)    ;SAVE ... FOR TYPEOUT
5338  030626  104402                          TYPOC                   ;TYPE OCTAL, ALL DIGITS
5339  030630  005711                  14$:    TST     (R1)            ;ANOTHER NUMBER ?
5340  030632  001403                          BEQ     7$              ;NO - EXIT
5341  030634  104401  030700                  TYPE    ,10$            ;TAB BETWEEN ELEMENTS
5342  030640  000771                          BR      6$              ;LOOP ON DATA TABLE VECTOR
5343  030642  104401  001177          7$:     TYPE    ,$CRLF          ;END THE LINE
5344  030646  012001                          MOV     (R0)+,R1        ;GET OPERAND LIST PTR
5345  030650  001406                          BEQ     15$             ;ALL DONE
5346  030652  012137  030662                  MOV     (R1)+,12$       ;OPERAND TITLE
5347  030656  001402                          BEQ     13$             ;SKIP IF ZERO
5348  030660  104401                          TYPE                    ;TYPE IT
5349  030662  000000                  12$:    .WORD   0               ;FROM HERE
5350  030664  000761                  13$:    BR      14$             ;GO FINISH DATA LIST
5351  030666  012601                  15$:    MOV     (SP)+,R1        ;RESTORE R1
5352  030670  012600                          MOV     (SP)+,R0        ;RESTORE R0
5353  030672  000207                          RTS     PC              ;RETURN
5354  030674  001206                  8$:     .WORD   $TESTN          ;
5355  030676  001124                  9$:     .WORD   $ERRPC          ;
5356  030700  000011                  10$:    .ASCIZ  <11>            ;<HT>
5357  030702  042524  052123  021440  11$:    .ASCIZ  "TEST # ERR PC "
5358  030710  042411  051122  050040
5359  030716  004503     000
5360          030722                          .EVEN
```

```
                                        TYPE ROUTINE

                                .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
                                ;*
                                
030722  105737  001165        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
030726  100002                        BPL     1$              ;;BR IF YES
030730  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
030732  000430                        BR      3$              ;;LEAVE
030734  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
030736  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
030742  122737  000001  001222        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
030750  001011                        BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
030752  132737  000100  001223        BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
030760  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
030762  010037  030772                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
030766  004737  031212                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
030772  000000                61$:    .WORD   0               ;;MESSAGE ADDRESS
030774  132737  000040  001223  62$:   BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
031002  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
031004  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
031006  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
031010  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
031012  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
031014  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
031020  000002                        RTI                     ;;RETURN
031022  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
031026  001430                        BEQ     8$
031030  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
031034  001006                        BNE     5$
031036  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
031040  104401                        TYPE                    ;;TYPE A CR AND LF
031042  001177                        $CRLF
031044  105037  031200                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
031050  000755                        BR      2$              ;;GET NEXT CHARACTER
031052  004737  031134        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
031056  123726  001164        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
031062  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
031064  013746  001162                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                              ;;AND THE NULL CHAR.
031070  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
031074  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
031076  004737  031134                JSR     PC,$TYPEC       ;;GO TYPE A NULL
031102  105337  031200                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

TYPE ROUTINE

```
5417  031106  000770                          BR      7$              ;;LOOP
5418
5419                                  ;HORIZONTAL TAB PROCESSOR
5420
5421  031110  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
5422  031114  004737  031134          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
5423  031120  132737  000007  031200          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
5424  031126  001372                          BNE     9$              ;;TAB STOP
5425  031130  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
5426  031132  000724                          BR      2$              ;;GET NEXT CHARACTER
5427  031134  105777  150016          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
5428  031140  100375                          BPL     $TYPEC
5429  031142  116677  000002  150010        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
5430  031150  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
5431  031156  001003                          BNE     1$              ;;BRANCH IF NO
5432  031160  105037  031200                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
5433  031164  000406                          BR      $TYPEX          ;;EXIT
5434  031166  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
5435  031174  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
5436  031176  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
5437  031200  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
5439  031202  000207          $TYPEX: RTS     PC
5439
```

```
                                       .SBTTL  APT COMMUNICATIONS ROUTINE
                                       ;;**************************************************************************
031204  112737  000001  031450 $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
031212  112737  000001  031446 $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
031220  000403                        BR      $ATYC
031222  112737  000001  031450 $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
031230                         $ATYC:
031230  010046                        MOV     R0,-(SP)          ;;PUSH R0 ON STACK
031232  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
031234  105737  031446                TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
031240  001450                        BEQ     5$                ;;IF NOT:  BR
031242  122737  000001  001222        CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
031250  001031                        BNE     3$                ;;IF NOT:  BR
031252  132737  000100  001223        BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
031260  001425                        BEQ     3$                ;;IF NOT:  BR
031262  017600  000004                MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
031266  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
031274  005737  001202          1$:   TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
031300  001375                        BNE     1$                ;;IF NOT:  WAIT
031302  010037  001216                MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
031306  105720                  2$:   TSTB    (R0)+             ;;FIND END OF MESSAGE
031310  001376                        BNE     2$
031312  163700  001216                SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
031316  006200                        ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
031320  010037  001220                MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
031324  012737  000004  001202        MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
031332  000413                        BR      5$
031334  017637  000004  031360  3$:   MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
031342  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
031350  013746  177776                MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
031354  004737  030722                JSR     PC,$TYPE          ;;CALL TYPE MACRO
031360  000000                  4$:   .WORD   0
031362                          5$:
031362  105737  031450          10$:  TSTB    $FFLG             ;;SHOULD REPORT FATAL ERROR?
031366  001416                        BEQ     12$               ;;IF NOT:  BR
031370  005737  001222                TST     $ENV              ;;RUNNING UNDER APT?
031374  001413                        BEQ     12$               ;;IF NOT:  BR
031376  005737  001202          11$:  TST     $MSGTYPE          ;;FINISHED LAST MESSAGE?
031402  001375                        BNE     11$               ;;IF NOT:  WAIT
031404  017637  000004  001204        MOV     @4(SP),$FATAL     ;;GET ERROR #
031412  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
031420  005237  001202                INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR
031424  105037  031450          12$:  CLRB    $FFLG             ;;CLEAR FATAL FLAG
031430  105037  031447                CLRB    $LFLG             ;;CLEAR LOG FLAG
031434  105037  031446                CLRB    $MFLG             ;;CLEAR MESSAGE FLAG
031440  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
031442  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
031444  000207                        RTS     PC                ;;RETURN
031446     000          $MFLG:  .BYTE   0                 ;;MESSG. FLAG
031447     000          $LFLG:  .BYTE   0                 ;;LOG FLAG
031450     000          $FFLG:  .BYTE   0                 ;;FATAL FLAG
           031452                  .EVEN
           000200          APTSIZE=200
           000001          APTENV=001
           000100          APTSPOOL=100
```

```
                                APT COMMUNICATIONS ROUTINE

  5496          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

031452  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
031456  116637  000001  031675        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
031464  112637  031677                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
031470  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
031474  000406                        BR      $TYPON
031476  112737  000001  031675 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
031504  112737  000006  031677        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
031512  112737  000005  031674 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
031520  010346                        MOV     R3,-(SP)        ;;SAVE R3
031522  010446                        MOV     R4,-(SP)        ;;SAVE R4
031524  010546                        MOV     R5,-(SP)        ;;SAVE R5
031526  113704  031677                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
031532  005404                        NEG     R4
031534  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
031540  110437  031676                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
031544  113704  031675                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
031550  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
031554  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
031556  006105          1$:   ROL     R5              ;;ROTATE MSB INTO "C"
031560  000404                        BR      3$              ;;GO DO MSB
031562  006105          2$:   ROL     R5              ;;FORM THIS DIGIT
031564  006105                        ROL     R5
031566  006105                        ROL     R5
031570  010503                        MOV     R5,R3
031572  006103          3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
031574  105337  031676                DECB    $OMODE          ;;TYPE THIS DIGIT?
031600  100016                        BPL     7$              ;;BR IF NO
031602  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
031606  001002                        BNE     4$              ;;TEST FOR 0
031610  005704                        TST     R4              ;;SUPPRESS THIS 0?
031612  001403                        BEQ     5$              ;;BR IF YES
```

BINARY TO OCTAL (ASCII) AND TYPE

```
5553  031614  005204                    4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
5554  031616  052703  000060                    BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
5555  031622  052703  000040            5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
5556  031626  110337  031672                    MOVB    R3,8$           ;;SAVE FOR TYPING
5557  031632  104401  031672                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
5558  031636  105337  031674            7$:     DECB    $OCNT           ;;COUNT BY 1
5559  031642  003347                            BGT     2$              ;;BR IF MORE TO DO
5560  031644  002402                            BLT     6$              ;;BR IF DONE
5561  031646  005204                            INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
5562  031650  000744                            BR      2$              ;;GO DO THE LAST DIGIT
5563  031652  012605                    6$:     MOV     (SP)+,R5        ;;RESTORE R5
5564  031654  012604                            MOV     (SP)+,R4        ;;RESTORE R4
5565  031656  012603                            MOV     (SP)+,R3        ;;RESTORE R3
5566  031660  016666  000002  000004           MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
5567  031666  012616                            MOV     (SP)+,(SP)
5568  031670  000002                            RTI                     ;;RETURN
5569  031672     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
5570  031673     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
5571  031674     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
5572  031675     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
5573  031676  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.

031700  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
031702  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
031706  005740                          TST     -(R0)           ;;BACKUP BY 2
031710  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
031712  006300                          ASL     R0              ;;POSITION FOR INDEXING
031714  016000  031734                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
031720  000200                          RTS     R0              ;;GO TO ROUTINE

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

031722  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
031724  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
031732  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
031734  031722                  $TRPAD: .WORD   $TRAP2
031736  030722                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
031740  031476                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
031742  031452                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
031744  031512                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
                                        POWER DOWN AND UP ROUTINES

                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
031746  012737  032120  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
031754  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
031762  010046                        MOV     R0,-(SP)         ;;PUSH R0 ON STACK
031764  010146                        MOV     R1,-(SP)         ;;PUSH R1 ON STACK
031766  010246                        MOV     R2,-(SP)         ;;PUSH R2 ON STACK
031770  010346                        MOV     R3,-(SP)         ;;PUSH R3 ON STACK
031772  010446                        MOV     R4,-(SP)         ;;PUSH R4 ON STACK
031774  010546                        MOV     R5,-(SP)         ;;PUSH R5 ON STACK
031776  017746  147144                MOV     @$WR,-(SP)       ;;PUSH @$WR ON STACK
032002  010637  032124                MOV     SP,$SAVR6        ;;SAVE SP
032006  012737  032020  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
032014  000000                        HALT
032016  000776                        BR      .-2              ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
032020  012737  032120  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
032026  013706  032124                MOV     $SAVR6,SP        ;;GET SP
032032  005037  032124                CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
032036  005237  032124          1$:   INC     $SAVR6           ;;WAIT FOR THE INC
032042  001375                        BNE     1$               ;;OF  WORD
032044  011600                        MOV     (SP),R0          ;GET SAVED $WR OFF STACK
032046  076600  000226                MED      226             ;RESTORE $WR CONTENTS
032052  012677  147070                MOV     (SP)+,@$WR       ;;POP STACK INTO @$WR
032056  012605                        MOV     (SP)+,R5         ;;POP STACK INTO R5
032060  012604                        MOV     (SP)+,R4         ;;POP STACK INTO R4
032062  012603                        MOV     (SP)+,R3         ;;POP STACK INTO R3
032064  012602                        MOV     (SP)+,R2         ;;POP STACK INTO R2
032066  012601                        MOV     (SP)+,R1         ;;POP STACK INTO R1
032070  012600                        MOV     (SP)+,R0         ;;POP STACK INTO R0
032072  012737  031746  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
032100  012737  000340  000026        MOV     #340,@#PWRVEC+2  ;;PRIO:7
032106  104401                        TYPE                     ;REPORT THE POWER FAILURE
032110  032126                $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
032112  012716                        MOV     (PC)+,(SP)       ;;RESTART AT START
032114  002734                $PWRAD: .WORD   START            ;;RESTART ADDRESS
032116  000002                        RTI
032120  000000                $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
032122  000776                        BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
032124  000000                $SAVR6: 0                        ;;PUT THE SP HERE
032126  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
032134  000122
                                        .EVEN
```

```
                                        .SBTTL  ERROR MESSAGES, DATA HEADERS, DATA VECTORS, OPERAND VECTORS, ETC

                                        ;MESSAGE PREFIX
032136  047510  035124  000040  ASCHOT: .ASCIZ  "HOT: "
032144  040527  046522  020072  ASCWRM: .ASCIZ  "WARM: "
032152     000
                                        ;ERROR MESSAGES HERE
032153     106  046455  042117  EMA:    .ASCIZ  "F-MODE EXERCISER - FPS ERROR"
032160  020105  054105  051105
032166  044503  042523  020122
032174  020055  050106  020123
032202  051105  047522  000122
032210  026504  047515  042504  EMB:    .ASCIZ  "D-MODE EXERCISER - FPS ERROR"
032216  042440  042530  041522
032224  051511  051105  026440
032232  043040  051520  042440
032240  051122  051117     000
032245     106  046455  042117  EMC:    .ASCIZ  "F-MODE EXERCISER - RESULT ERROR"
032252  020105  054105  051105
032260  044503  042523  020122
032266  020055  042522  052523
032274  052114  042440  051122
032302  051117     000
032305     104  046455  042117  EMD:    .ASCIZ  "D-MODE EXERCISER - RESULT ERROR"
032312  020105  054105  051105
032320  044503  042523  020122
032326  020055  042522  052523
032334  052114  042440  051122
032342  051117     000
032345     101  042104  043050  EME:    .ASCIZ  "ADD(F/D) - RESULT ERROR"
032352  042057  020051  020055
032360  042522  052523  052114
032366  042440  051122  051117
032374     000
032375     123  041125  043050  EMF:    .ASCIZ  "SUB(F/D) - RESULT ERROR"
032402  042057  020051  020055
032410  042522  052523  052114
032416  042440  051122  051117
032424     000
032425     115  046125  043050  EMG:    .ASCIZ  "MUL(F/D) - RESULT ERROR"
032432  042057  020051  020055
032440  042522  052523  052114
032446  042440  051122  051117
032454     000
032455     104  053111  043050  EMH:    .ASCIZ  "DIV(F/D) - RESULT ERROR"
032462  042057  020051  020055
032470  042522  052523  052114
032476  042440  051122  051117
032504     000
032505     125  042516  050130  EMI:    .ASCIZ  "UNEXPECTED FLOATING POINT TRAP, IGNORED AND CONTINUING"
032512  041505  042524  020104
032520  046106  040517  044524
032526  043516  050040  044517
032534  052116  052040  040522
032542  026120  044440  047107
```

```
 5714  032550  051117  042105  040440
 5715  032556  042116  041440  047117
 5716  032564  044524  052516  047111
 5717  032572  000107
 5718  032574  042101  043104  026440  EMJ:    .ASCIZ  "ADDF - FPS ERROR"
 5719  032602  043040  051520  042440
 5720  032610  051122  051117     000
 5721  032615     123  041125  020106  EMK:    .ASCIZ  "SUBF - FPS ERROR"
 5722  032622  020055  050106  020123
 5723  032630  051105  047522  000122
 5724  032636  052515  043114  026440  EML:    .ASCIZ  "MULF - FPS ERROR"
 5725  032644  043040  051520  042440
 5726  032652  051122  051117     000
 5727  032657     104  053111  020106  EMM:    .ASCIZ  "DIVF - FPS ERROR"
 5728  032664  020055  050106  020123
 5729  032672  051105  047522  000122
 5730  032700  042101  042104  026440  EMN:    .ASCIZ  "ADDD - FPS ERROR"
 5731  032706  043040  051520  042440
 5732  032714  051122  051117     000
 5733  032721     123  041125  020104  EMO:    .ASCIZ  "SUBD - FPS ERROR"
 5734  032726  020055  050106  020123
 5735  032734  051105  047522  000122
 5736  032742  052515  042114  026440  EMP:    .ASCIZ  "MULD - FPS ERROR"
 5737  032750  043040  051520  042440
 5738  032756  051122  051117     000
 5739  032763     104  053111  020104  EMQ:    .ASCIZ  "DIVD - FPS ERROR"
 5740  032770  020055  050106  020123
 5741  032776  051105  047522  000122
 5742  033004  042101  043104  026440  EMR:    .ASCIZ  "ADDF - FEC/FEA ERROR"
 5743  033012  043040  041505  043057
 5744  033020  040505  042440  051122
 5745  033026  051117     000
 5746  033031     123  041125  020106  EMS:    .ASCIZ  "SUBF - FEC/FEA ERROR"
 5747  033036  020055  042506  027503
 5748  033044  042506  020101  051105
 5749  033052  047522  000122
 5750  033056  052515  043114  026440  EMT:    .ASCIZ  "MULF - FEC/FEA ERROR"
 5751  033064  043040  041505  043057
 5752  033072  040505  042440  051122
 5753  033100  051117     000
 5754  033103     104  053111  020106  EMU:    .ASCIZ  "DIVF - FEC/FEA ERROR"
 5755  033110  020055  042506  027503
 5756  033116  042506  020101  051105
 5757  033124  047522  000122
 5758  033130  042101  042104  026440  EMV:    .ASCIZ  "ADDD - FEC/FEA ERROR"
 5759  033136  043040  041505  043057
 5760  033144  040505  042440  051122
 5761  033152  051117     000
 5762  033155     123  041125  020104  EMW:    .ASCIZ  "SUBD - FEC/FEA ERROR"
 5763  033162  020055  042506  027503
 5764  033170  042506  020101  051105
 5765  033176  047522  000122
 5766  033202  052515  042114  026440  EMX:    .ASCIZ  "MULD - FEC/FEA ERROR"
 5767  033210  043040  041505  043057
 5768  033216  040505  042440  051122
 5769  033224  051117     000
```

```
                                        ERROR MESSAGES, DATA HEADERS, DATA VECTORS, OPERAND VECTORS, ETC

  5770  033227     104  053111  020104  EMY:    .ASCIZ  "DIVD - FEC/FEA ERROR"
  5771  033234  020055  042506  027503
  5772  033242  042506  020101  051105
  5773  033250  047522  000122
  5774
  5775
  5776                                          ;DATA HEADERS HERE
  5777  033254  054105  023520  004504  DHA:    .ASCIZ  "EXP'D  RCV'D"
  5778  033262  041522  023526  000104
  5779  033270  026455  042455  050130  DHB:    .ASCIZ  "---EXPECTED--- ---RECEIVED---"
  5780  033276  041505  042524  026504
  5781  033304  026455  026411  026455
  5782  033312  042522  042503  053111
  5783  033320  042105  026455  000055
  5784  033326  026455  026455  026455  DHC:    .ASCIZ  "----------EXPECTED---------- ----------RECEIVED----------"
  5785  033334  026455  026455  042455
  5786  033342  050130  041505  042524
  5787  033350  026504  026455  026455
  5788  033356  026455  026455  026455
  5789  033364  026411  026455  026455
  5790  033372  026455  026455  026455
  5791  033400  042522  042503  053111
  5792  033406  042105  026455  026455
  5793  033414  026455  026455  026455
  5794  033422  000055
  5795  033424  046117  020104  041520  DHD:    .ASCIZ  "OLD PC OLD PS   FPS     FEC     FEA     $FPS    $FEC    $FEA"
  5796  033432  047411  042114  050040
  5797  033440  004523  043040  051520
  5798  033446  020011  042506  004503
  5799  033454  043040  040505  020011
  5800  033462  043044  051520  020011
  5801  033470  043044  041505  020011
  5802  033476  043044  040505     000
  5803  033503     105  050130  042047  DHE:    .ASCIZ  "EXP'D-FEC-RCV'D        EXP'D-FEA-RCV'D"
  5804  033510  043055  041505  051055
  5805  033516  053103  042047  042411
  5806  033524  050130  042047  043055
  5807  033532  040505  051055  053103
  5808  033540  042047     000
  5809
  5810
  5811                                          ;DATA VECTORS HERE
  5812          033544                          .EVEN
  5813  033544  002400  002362  000000  DTA:    .WORD   $FPS,FPS,0
  5814  033552  002402  002364  002404  DTB:    .WORD   $FEC,FEC,$FEA,FEA,0
  5815  033560  002366  000000
  5816  033564  002416  002420  002406  DTD:    .WORD   ANS2,ANS2+2,ANS1,ANS1+2,0
  5817  033572  002410  000000
  5818  033576  002416  002420  002422  DTE:    .WORD   ANS2,ANS2+2,ANS2+4,ANS2+6
  5819  033604  002424
  5820  033606  002406  002410  002412          .WORD   ANS1,ANS1+2,ANS1+4,ANS1+6,0
  5821  033614  002414  000000
  5822  033620  002370  002372  002362  DTF:    .WORD   FPPOPC,FPPOPS,FPS,FEC,FEA,$FPS,$FEC,$FEA,0
  5823  033626  002364  002366  002400
  5824  033634  002402  002404  000000
  5825
```

```
                                                        ;OPERAND VECTORS HERE
                                                        .EVEN
033642  034102  002426  002430  LOD:    .WORD   XLO,LONUM,LONUM+2,LONUM+4,LONUM+6,0
033650  002432  002434  000000
033656  034112  002436  002440  HID:    .WORD   XHI,HINUM,HINUM+2,HINUM+4,HINUM+6,0
033664  002442  002444  000000
033672  034102  002426  002430  LOF:    .WORD   XLO,LONUM,LONUM+2,0
033700  000000
033702  034112  002436  002440  HIF:    .WORD   XHI,HINUM,HINUM+2,0
033710  000000
033712  034122  002446  002450  OP1F:   .WORD   XOP1,OP1,OP1+2,0
033720  000000
033722  034130  002456  002460  OP2F:   .WORD   XOP2,OP2,OP2+2,0
033730  000000
033732  034136  002466  002470  OP3F:   .WORD   XOP3,OP3,OP3+2,0
033740  000000
033742  034144  002476  002500  OP4F:   .WORD   XOP4,OP4,OP4+2,0
033750  000000
033752  034160  002506  002510  OP5F:   .WORD   XOP5,OP5,OP5+2,0
033760  000000
033762  034152  002516  002520  OP6F:   .WORD   XOP6,OP6,OP6+2,0
033770  000000
033772  034122  002446  002450  OP1D:   .WORD   XOP1,OP1,OP1+2,OP1+4,OP1+6,0
034000  002452  002454  000000
034006  034130  002456  002460  OP2D:   .WORD   XOP2,OP2,OP2+2,OP2+4,OP2+6,0
034014  002462  002464  000000
034022  034136  002466  002470  OP3D:   .WORD   XOP3,OP3,OP3+2,OP3+4,OP3+6,0
034030  002472  002474  000000
034036  034144  002476  002500  OP4D:   .WORD   XOP4,OP4,OP4+2,OP4+4,OP4+6,0
034044  002502  002504  000000
034052  034160  002506  002510  OP5D:   .WORD   XOP5,OP5,OP5+2,OP5+4,OP5+6,0
034060  002512  002514  000000
034066  034152  002516  002520  OP6D:   .WORD   XOP6,OP6,OP6+2,OP6+4,OP6+6,0
034074  002522  002524  000000


                                                        ;OPERAND TITLES
034102  047514  052516  035115  XLO:    .ASCIZ  "LONUM:"<11>
034110  000011
034112  044510  052516  035115  XHI:    .ASCIZ  "HINUM:"<11>
034120  000011
034122  050117  035061  000011  XOP1:   .ASCIZ  "OP1:"<11>
034130  050117  035062  000011  XOP2:   .ASCIZ  "OP2:"<11>
034136  050117  035063  000011  XOP3:   .ASCIZ  "OP3:"<11>
034144  050117  035064  000011  XOP4:   .ASCIZ  "OP4:"<11>
034152  050117  035066  000011  XOP6:   .ASCIZ  "OP6:"<11>
034160  050117  035065  000011  XOP5:   .ASCIZ  "OP5:"<11>
                                        ;THE END
        000001                          .END
```

```
ABASE = 000000            993
ACOM1 = 000000            993
ACOM2 = 000000            993
ACPUOP= 000000            993    1008
ADDC0   002526           1197#   1305    4030    4329#
ADDC1   002530           1198#   4030    4373#
ADDC2   002532           1199#   4030    4352#
ADDC3   002534           1200#   4030    4375#
ADDC4   002536           1201#   4030    4418#
ADDC5   002540           1202#   4030    4479#   4499#
ADDC6   002542           1203#   4030    4482#   4502#
ADDC7   002544           1204#   4030    4513#   4558#
ADDC8   002546           1205#   4030    4559#
ADDM0 = 000000            993
ADDM1 = 000000            993
ADDM10= 000000            993
ADDM11= 000000            993
ADDM12= 000000            993
ADDM13= 000000            993
ADDM14= 000000            993
ADDM15= 000000            993
ADDM2 = 000000            993
ADDM3 = 000000            993
ADDM4 = 000000            993
ADDM5 = 000000            993
ADDM6 = 000000            993
ADDM7 = 000000            993
ADDM8 = 000000            993
ADDM9 = 000000            993
ADEVCT= 000000            993     999
ADEVM = 000000            993
AEND1   004036           1439    1446    1450#
AEND10  006412           1864    1873    1880    1884#
AEND11  006676           1937    1944    1948#
AEND12  007172           2001    2010    2014#
AEND13  007420           2055    2062    2066#
AEND14  007656           2107    2116    2120#
AEND15  010164           2173    2180    2185    2189#
AEND16  010512           2242    2251    2258    2262#
AEND17  010762           2303    2310    2315    2319#
AEND2   004332           1503    1512    1516#
AEND20  011252           2360    2369    2376    2380#
AEND3   004560           1557    1564    1568#
AEND4   005016           1609    1618    1622#
AEND5   005324           1675    1682    1687    1691#
AEND6   005652           1745    1754    1761    1765#
AEND7   006122           1807    1814    1819    1823#
AENV  = 000000            993    1004
AENVM = 000000            993    1005
AERR1   003736           1421    1425#
AERR11  006576           1919    1923#
AERR12  007062           1983    1987#
AERR15  010042           2155    2159#
AERR16  010350           2224    2228#
AERR2   004222           1485    1489#
AERR5   005202           1657    1661#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
AERR6   005510          1727    1731#
AFATAL= 000000           993     996
AMAOR1= 000000           993
AMAOR2= 000000           993
AMAOR3= 000000           993
AMAOR4= 000000           993
AMAMS1= 000000           993
AMAMS2= 000000           993
AMAMS3= 000000           993
AMAMS4= 000000           993
AMSGAD= 000000           993    1001
AMSGLG= 000000           993    1002
AMSGTY= 000000           993     995
AMTYP1= 000000           993
AMTYP2= 000000           993
AMTYP3= 000000           993
AMTYP4= 000000           993
ANS1    002406          1175#   1422#   1441#   1442#   1443#   1444    1447#   1486#   1505#   1506#   1507#   1508#   1509#
                        1510    1513#   1552#   1559#   1560#   1561#   1562    1565#   1604#   1611#   1612#   1613#   1614#
                        1615#   1616    1619#   1658#   1677#   1678#   1679#   1680    1683#   1684#   1685    1688#   1728#
                        1747#   1748#   1749#   1750#   1751#   1752    1755#   1756#   1757#   1758#   1759    1762#   1802#
                        1809#   1810#   1811#   1812    1815#   1816#   1817    1820#   1859#   1866#   1867#   1868#   1869#
                        1870#   1871    1874#   1875#   1876#   1877#   1878    1881#   1920#   1939#   1940#   1941#   1942
                        1945#   1984#   2003#   2004#   2005#   2006#   2007#   2008    2011#   2050#   2057#   2058#   2059#
                        2060    2063#   2102#   2109#   2110#   2111#   2112#   2113#   2114    2117#   2156#   2175#   2176#
                        2177#   2178    2181#   2182#   2183    2186#   2225#   2244#   2245#   2246#   2247#   2248#   2249
                        2252#   2253#   2254#   2255#   2256    2259#   2298#   2305#   2306#   2307#   2308    2311#   2312#
                        2313    2316#   2355#   2362#   2363#   2364#   2365#   2366#   2367    2370#   2371#   2372#   2373#
                        2374    2377#   2416#   2435#   2436#   2437#   2438    2441#   2480#   2499#   2500#   2501#   2502#
                        2503#   2504    2507#   2546#   2553#   2554#   2555#   2556    2559#   2598#   2605#   2606#   2607#
                        2608#   2609#   2610    2613#   2651#   2670#   2671#   2672#   2673    2676#   2677#   2678    2681#
                        2720#   2739#   2740#   2741#   2742#   2743#   2744    2747#   2748#   2749#   2750#   2751    2754#
                        2793#   2800#   2801#   2802#   2803    2806#   2807#   2808    2811#   2850#   2857#   2858#   2859#
                        2860#   2861#   2862    2865#   2866#   2867#   2868#   2869    2872#   2911#   2930#   2931#   2932#
                        2933    2936#   2975#   2994#   2995#   2996#   2997#   2998#   2999    3002#   3041#   3060#   3061#
                        3062#   3063    3066#   3105#   3124#   3125#   3126#   3127#   3128#   3129    3132#   3171#   3190#
                        3191#   3192#   3193    3196#   3197#   3198    3201#   3240#   3259#   3260#   3261#   3262#   3263#
                        3264    3267#   3268#   3269#   3270#   3271    3274#   3313#   3332#   3333#   3334#   3335    3338#
                        3339#   3340    3343#   3382#   3401#   3402#   3403#   3404#   3405#   3406    3409#   3410#   3411#
                        3412#   3413    3416#   3455#   3474#   3475#   3476#   3477    3480#   3518#   3537#   3538#   3539#
                        3540#   3541#   3542    3545#   3584#   3603#   3604#   3605#   3606    3609#   3610#   3611    3614#
                        3652#   3671#   3672#   3673#   3674#   3675#   3676    3679#   3680#   3681#   3682#   3683    3686#
                        3745#   3753#   3754#   3755#   3756    3760#   3761#   3762    3766#   3767#   3768    3772#   3773#
                        3774    3778#   3779#   3780    3784#   3785#   3786    3790#   3852#   3860#   3861#   3862#   3863#
                        3864#   3865    3869#   3870#   3871#   3872#   3873    3877#   3878#   3879#   3880#   3881    3885#
                        3886#   3887#   3888#   3889    3893#   3894#   3895#   3896#   3897    3901#   3902#   3903#   3904#
                        3905    3909#   5816    5820
ANS2    002416          1177#   1405    1411    1469    1475    1535    1541    1587    1593    1641    1647    1711    1717
                        1785    1791    1842    1848    1903    1909    1967    1973    2033    2039    2085    2091    2139
                        2145    2208    2214    2281    2287    2338    2344    2399    2405    2463    2469    2529    2535
                        2581    2587    2634    2640    2703    2709    2776    2782    2833    2839    2894    2900    2958
                        2964    3024    3030    3088    3094    3154    3160    3223    3229    3296    3302    3365    3371
                        3438    3444    3501    3507    3567    3573    3635    3641    3724    3727    3831    3834    5816
                        5818
APASS = 000000           993     998
APRIOR= 000000           993
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
APTCSU= 000040           5391    5496#
APTENV= 000001           5252    5384    5452    5494#
APTSIZ= 000200           1295    5493#
APTSP0= 000100           5386    5454    5495#
ARET1   003712           1392    1418#
ARET10  006262           1829    1855#
ARET11  006552           1890    1916#
ARET12  007036           1954    1980#
ARET13  007332           2020    2046#
ARET14  007560           2072    2098#
ARET15  010016           2126    2152#
ARET16  010324           2195    2221#
ARET17  010652           2268    2294#
ARET2   004176           1456    1482#
ARET20  011122           2325    2351#
ARET3   004472           1522    1548#
ARET4   004720           1574    1600#
ARET5   005156           1628    1654#
ARET6   005464           1698    1724#
ARET7   006012           1772    1798#
ASCHOT  032136           1384    5661#
ASCWRM  032144           1386    5662#
ASWREG= 000000            993    1006
ATESTN= 000000            993     997
ATST1   003774           1426    1434    1437#
ATST10  006306           1858    1862#
ATST11  006634           1924    1932    1935#
ATST12  007120           1988    1996    1999#
ATST13  007356           2049    2053#
ATST14  007604           2101    2105#
ATST15  010100           2160    2168    2171#
ATST16  010406           2229    2237    2240#
ATST17  010676           2297    2301#
ATST2   004260           1490    1498    1501#
ATST20  011146           2354    2358#
ATST3   004516           1551    1555#
ATST4   004744           1603    1607#
ATST5   005240           1662    1670    1673#
ATST6   005546           1732    1740    1743#
ATST7   006036           1801    1805#
AUNIT = 000000            993    1000
AUSWR = 000000            993    1007
AVECT1= 000000            993
AVECT2= 000000            993
BGAMES  002622           1233#   1310
BIT0  = 000001            862#
BIT00 = 000001            852#    862
BIT01 = 000002            851#    861
BIT02 = 000004            850#    860
BIT03 = 000010            849#    859
BIT04 = 000020            848#    858    1316    1366    3930
BIT05 = 000040            847#    857
BIT06 = 000100            846#    856
BIT07 = 000200            845#    855
BIT08 = 000400            844#    854    5179
BIT09 = 001000            843#    853    5187    5262
```

```
BIT1   = 000002            861#
BIT10  = 002000            842#  5240
BIT11  = 004000            841#  5194
BIT12  = 010000            840#  1348   1376   1378   1381   3936
BIT13  = 020000            839#  5247
BIT14  = 040000            838#  5165
BIT15  = 100000            837#
BIT2   = 000004            860#
BIT3   = 000010            859#
BIT4   = 000020            858#
BIT5   = 000040            857#
BIT6   = 000100            856#
BIT7   = 000200            855#
BIT8   = 000400            854#
BIT9   = 001000            853#
BPTVEC= 000014             869#
CR     = 000015            777#  5430   5440
CRLF   = 000200            778#  5401   5440
DDISP  = 177570            784#   974   1283
DEND1   014376            2928   2935   2939#
DEND10  017142            3394   3399   3408   3415   3419#
DEND11  017426            3472   3479   3483#
DEND12  017722            3535   3544   3548#
DEND13  020230            3601   3608   3613   3617#
DEND14  020556            3669   3678   3685   3689#
DEND2   014672            2992   3001   3005#
DEND3   015156            3058   3065   3069#
DEND4   015452            3122   3131   3135#
DEND5   015760            3188   3195   3200   3204#
DEND6   016306            3257   3266   3273   3277#
DEND7   016614            3330   3337   3342   3346#
DERR1   014276            2910   2914#
DERR10  017000            3381   3385#
DERR11  017326            3454   3458#
DERR12  017612            3517   3521#
DERR13  020106            3583   3587#
DERR14  020414            3651   3655#
DERR2   014562            2974   2978#
DERR3   015056            3040   3044#
DERR4   015342            3104   3108#
DERR5   015636            3170   3174#
DERR6   016144            3239   3243#
DERR7   016472            3312   3316#
DHA     033254            1033   1037   1041   1089   1093   1097   1101   1105   1143   1147   5777#
DHB     033270            1051   1059   1067   1075   1152   5779#
DHC     033326            1055   1063   1071   1079   1156   5784#
DHD     033424            1084   5795#
DHE     033503            1110   1114   1118   1122   1126   1130   1134   1138   5803#
DISPLA  001150             974#  1283*  1291*  5209*  5239*
DISPRE  000174             906#  1291
DIVC0   002564            1214#  4037   4920*
DIVC1   002566            1215#  4037   4948*
DIVC2   002570            1216#  4037   4933*
DIVC3   002572            1217#  4037   5035*  5056*
DIVC4   002574            1218#  4037   5038*  5059*
DIVC5   002576            1219#  4037   5024*
```

```
DIVC6   002600            1220#   4037    5027#
DRET1   014252            2881    2907#
DRET10  016754            3352    3378#
DRET11  017302            3425    3451#
DRET12  017566            3488    3514#
DRET13  020062            3554    3580#
DRET14  020370            3622    3648#
DRET2   014536            2945    2971#
DRET3   015032            3011    3037#
DRET4   015316            3075    3101#
DRET5   015612            3141    3167#
DRET6   016120            3210    3236#
DRET7   016446            3283    3309#
DSWR  = 177570             783#    973    1282
DTA     033544            1033    1037    1041    1089    1093    1097    1101    1105    1143    1147    5813#
DTB     033552            1110    1114    1118    1122    1126    1130    1134    1138    5814#
DTD     033564            1051    1059    1067    1075    1152    5816#
DTE     033576            1055    1063    1071    1079    1156    5818#
DTF     033620            1084    5822#
DTST1   014334            2915    2923    2926#
DTST10  017036            3386    3397#
DTST11  017364            3459    3467    3470#
DTST12  017650            3522    3530    3533#
DTST13  020144            3588    3596    3599#
DTST14  020452            3656    3664    3667#
DTST2   014620            2979    2987    2990#
DTST3   015114            3045    3053    3056#
DTST4   015400            3109    3117    3120#
DTST5   015674            3175    3183    3186#
DTST6   016202            3244    3252    3255#
DTST7   016530            3317    3325    3328#
EEND1   021226            3750    3758    3764    3770    3776    3782    3788    3793#
EEND2   021766            3857    3867    3875    3883    3891    3899    3907    3912#
EMA     032153            1143    5666#
EMB     032210            1147    5671#
EMC     032245            1152    5676#
EMD     032305            1156    5682#
EME     032345            1051    1055    5688#
EMF     032375            1059    1063    5693#
EMG     032425            1067    1071    5698#
EMH     032455            1075    1079    5703#
EMI     032505            1084    5708#
EMJ     032574            1033    5718#
EMK     032615            1037    5721#
EML     032636            1041    5724#
EMM     032657            1089    5727#
EMN     032700            1093    5730#
EMO     032721            1097    5733#
EMP     032742            1101    5736#
EMQ     032763            1105    5739#
EMR     033004            1110    5742#
EMS     033031            1114    5746#
EMT     033056            1118    5750#
EMTVEC= 000030             872#   1266*   1267*
EMU     033103            1122    5754#
EMV     033130            1126    5758#
```

```
EMV001  001232          1033#
EMV002  001256          1037#
EMV003  001302          1041#
EMV004  001326          1046#
EMV005  001352          1051#
EMV006  001376          1055#
EMV007  001422          1059#
EMV010  001446          1063#
EMV011  001472          1067#
EMV012  001516          1071#
EMV013  001542          1075#
EMV014  001566          1079#
EMV015  001612          1084#
EMV016  001636          1089#
EMV017  001662          1093#
EMV020  001706          1097#
EMV021  001732          1101#
EMV022  001756          1105#
EMV023  002002          1110#
EMV024  002026          1114#
EMV025  002052          1118#
EMV026  002076          1122#
EMV027  002122          1126#
EMV030  002146          1130#
EMV031  002172          1134#
EMV032  002216          1138#
EMV033  002242          1143#
EMV034  002266          1147#
EMV035  002312          1152#
EMV036  002336          1156#
EMW     033155          1130    5762#
EMX     033202          1134    5766#
EMY     033227          1138    5770#
EREG0   002602          1223#   5228*
EREG1   002604          1224#   5229*
EREG2   002606          1225#   5230*
EREG3   002610          1226#   5231*
EREG4   002612          1227#   5232*
EREG5   002614          1228#   5233*
EREG6   002616          1229#   5234*   5235*
EREG7   002620          1230#   5236*
ERRVEC= 000004           865#   1280    1281*   1292*   5170    5171*   5173*   5176*
ETST1   021032          3744    3748#
ETST2   021512          3851    3855#
EXPFEA  002376          1171#   1392*   1456*   1522*   1574*   1628*   1698*   1772*   1829*   1890*   1954*   2020*   2072*
                        2126*   2195*   2268*   2325*   2386*   2450*   2516*   2568*   2621*   2690*   2763*   2820*   2881*
                        2945*   3011*   3075*   3141*   3210*   3283*   3352*   3425*   3488*   3554*   3622*   4485    4505
                        4562    4743    4754    4785    4836    5030    5041    5062
FEA     002366          1167#   1398*   1433    1462*   1497    1528*   1580*   1634*   1669    1704*   1739    1778*   1835*
                        1896*   1931    1960*   1995    2026*   2078*   2132*   2167    2201*   2236    2274*   2331*   2392*
                        2427    2456*   2491    2522*   2574*   2627*   2662    2696*   2731    2769*   2826*   2887*   2922
                        2951*   2986    3017*   3052    3081*   3116    3147*   3182    3216*   3251    3289*   3324    3358*
                        3393    3431*   3466    3494*   3529    3560*   3595    3628*   3663    3701*   3806*   5814    5822
FEC     002364          1166#   1397*   1427*   1429    1461*   1491*   1493    1527*   1579*   1633*   1663*   1665    1703*
                        1733*   1735    1777*   1834*   1895*   1925*   1927    1959*   1989*   1991    2025*   2077*   2131*
                        2161*   2163    2200*   2230*   2232    2273*   2330*   2391*   2421*   2423    2455*   2485*   2487
```

```
                        2521*  2573*  2626*  2656*  2658   2695*  2725*  2727   2768*  2825*  2886*  2916*  2918
                        2950*  2980*  2982   3016*  3046*  3048   3080*  3110*  3112   3146*  3176*  3178   3215*
                        3245*  3247   3288*  3318*  3320   3357*  3387*  3389   3430*  3460*  3462   3493*  3523*
                        3525   3559*  3589*  3591   3627*  3657*  3659   3700*  3805*  4135*  5814   5822
FPPILT  023330          1301   4132*
FPPOPC  002370          1168*  4132*  4145   5822
FPPOPS  002372          1169*  4133*  4144   5822
FPPOSP  002374          1170*
FPPVEC= 000244           887*  1301*  1302*
FPS     002362          1165*  1396*  1419*  1420   1450*  1460*  1483*  1484   1516*  1526*  1549*  1550   1568*
                        1578*  1601*  1602   1622*  1632*  1655*  1656   1691*  1702*  1725*  1726   1765*  1776*
                        1799*  1800   1823*  1833*  1856*  1857   1884*  1894*  1917*  1918   1948*  1958*  1981*
                        1982   2014*  2024*  2047*  2048   2066*  2076*  2099*  2100   2120*  2130*  2153*  2154
                        2189*  2199*  2222*  2223   2262*  2272*  2295*  2296   2319*  2329*  2352*  2353   2380*
                        2390*  2413*  2414   2444*  2454*  2477*  2478   2510*  2520*  2543*  2544   2562*  2572*
                        2595*  2596   2616*  2625*  2648*  2649   2684*  2694*  2717*  2718   2757*  2767*  2790*
                        2791   2814*  2824*  2847*  2848   2875*  2885*  2908*  2909   2939*  2949*  2972*  2973
                        3005*  3015*  3038*  3039   3069*  3079*  3102*  3103   3135*  3145*  3168*  3169   3204*
                        3214*  3237*  3238   3277*  3287*  3310*  3311   3346*  3356*  3379*  3380   3419*  3429*
                        3452*  3453   3483*  3492*  3515*  3516   3548*  3558*  3581*  3582   3617*  3626*  3649*
                        3650   3689*  3699*  3742*  3743   3793*  3804*  3849*  3850   3912*  4134*  4136   4139
                        5813   5822
GNS     = ****** U       905   5605   5606   5607   5608
HID     033656          1055   1063   1071   1079   5830*
HIF     033702          1051   1059   1067   1075   5834*
HINUM   002436          1181*  1400   1403   1410   1464   1467   1474   1530   1533   1540   1582   1585   1592
                        1636   1639   1646   1706   1709   1716   1780   1783   1790   1837   1840   1847   1898
                        1901   1908   1962   1965   1972   2028   2031   2038   2080   2083   2090   2134   2137
                        2144   2203   2206   2213   2276   2279   2286   2333   2336   2343   2394   2397   2404
                        2458   2461   2468   2524   2527   2534   2576   2579   2586   2629   2632   2639   2698
                        2701   2708   2771   2774   2781   2828   2831   2838   2889   2892   2899   2953   2956
                        2963   3019   3022   3029   3083   3086   3093   3149   3152   3159   3218   3221   3228
                        3291   3294   3301   3360   3363   3370   3433   3436   3443   3496   3499   3506   3562
                        3565   3572   3630   3633   3640   5830   5834
HOTWRM  030464          1384*  1386*  5300*
HT      = 000011         775*  5399   5440
IOTVEC= 000020           870*  1264*  1265*
LF      = 000012         776*  5434   5440
LOD     033642          1055   1063   1071   1079   5828*
LOF     033672          1051   1059   1067   1075   5832*
LONUM   002426          1179*  1400   1402   1409   1464   1466   1473   1530   1532   1539   1582   1584   1591
                        1636   1638   1645   1706   1708   1715   1780   1782   1789   1837   1839   1846   1898
                        1900   1907   1962   1964   1971   2028   2030   2037   2080   2082   2089   2134   2136
                        2143   2203   2205   2212   2276   2278   2285   2333   2335   2342   2394   2396   2403
                        2458   2460   2467   2524   2526   2533   2576   2578   2585   2629   2631   2638   2698
                        2700   2707   2771   2773   2780   2828   2830   2837   2889   2891   2898   2953   2955
                        2962   3019   3021   3028   3083   3085   3092   3149   3151   3158   3218   3220   3227
                        3291   3293   3300   3360   3362   3369   3433   3435   3442   3496   3498   3505   3562
                        3564   3571   3630   3632   3639   5828   5832
MED     = 076600         879*  1315   1365   1369   1379   3929   3937   3943   5637
MEND1   011536          2433   2440   2444*
MEND10  014112          2855   2864   2871   2875*
MEND2   012032          2497   2506   2510*
MEND3   012260          2551   2558   2562*
MEND4   012516          2603   2612   2616*
MEND5   013024          2668   2675   2680   2684*
```

```
MEND6   013352            2737    2746    2753    2757#
MEND7   013622            2798    2805    2810    2814#
MERR1   011436            2415    2419#
MERR2   011722            2479    2483#
MERR5   012702            2650    2654#
MERR6   013210            2719    2723#
MRET1   011412            2386    2412#
MRET10  013762            2820    2846#
MRET2   011676            2450    2476#
MRET3   012172            2516    2542#
MRET4   012420            2568    2594#
MRET5   012656            2621    2647#
MRET6   013164            2690    2716#
MRET7   013512            2763    2789#
MTST1   011474            2420    2428    2431#
MTST10  014006            2849    2853#
MTST2   011760            2484    2492    2495#
MTST3   012216            2545    2549#
MTST4   012444            2597    2601#
MTST5   012740            2655    2663    2666#
MTST6   013246            2724    2732    2735#
MTST7   013536            2792    2796#
MULC0   002550            1207#   4034    4661*
MULC1   002552            1208#   4034    4692*
MULC2   002554            1209#   4034    4748*   4779*
MULC3   002556            1210#   4034    4751*   4782*
MULC4   002560            1211#   4034    4737*
MULC5   002562            1212#   4034    4740*
NEWPAS  003404            1319    1321    1346#   3987
NWPAS1  002723            1245#   1353
OP1     002446            1183#   3704    3713    3734    3809    3820    3841    5836    5848
OP1D    033772            1156    5848#
OP1F    033712            1152    5836#
OP2     002456            1185#   3706    3714    3735    3811    3825    3844    5838    5850
OP2D    034006            1156    5850#
OP2F    033722            1152    5838#
OP3     002466            1187#   3708    3716    3738    3811    3821    3842    5840    5852
OP3D    034022            1156    5852#
OP3F    033732            1152    5840#
OP4     002476            1189#   3704    3717    3736    3809    3826    3845    5842    5854
OP4D    034036            1156    5854#
OP4F    033742            1152    5842#
OP5     002506            1191#   3706    3722    3740    3813    3814    3823    3843    5844    5856
OP5D    034052            1156    5856#
OP5F    033752            1152    5844#
OP6     002516            1193#   3708    3709    3718    3737    3813    3816    3828    3846    5846    5858
OP6D    034066            1156    5858#
OP6F    033762            1152    5846#
PIRQ  = 177772             782#
PIRQVE= 000240             876#
PR0   = 000000             799#
PR1   = 000040             800#
PR2   = 000100             801#
PR3   = 000140             802#
PR4   = 000200             803#
PR5   = 000240             804#
```

```
PR6    = 000300            805#
PR7    = 000340            806#
PS     = 177776            779#    780
PSW    = 177776            780#
PWRVEC= 000024             871#   1270#   1271#   5615#   5616#   5625#   5631#   5645#   5646#
RAND    023422            4158    4161#
RANDL2  023414            1399    1529    1635    1779    1897    2027    2133    2275    2393    2523    2628    2770    2888
                          3018    3148    3290    3432    3561    3703    3705    3707    4159#
RANDL4  023404            1463    1581    1705    1836    1961    2079    2202    2332    2457    2575    2697    2827    2952
                          3082    3217    3359    3495    3629    3808    3810    3812    4156#
RESVEC= 000010             866#
RFLAG  = 000144            883#   1369    3937
RMAMI= 000022              881#   1315    1365    3929
SCRLF   022366            4025    4043#
SHDR1   022371            4002    4044#
SHDR2   022736            4006    4085#
SHDR3   023127            4010    4106#
SHT     022364            4023    4042#
STACK  = 001100            770#   1262    1346
START   002734             909    1255#   5650
STATS   022174            3977    3998#
STKLMT= 177774             781#
SUBPAS  003452            1349    1351    1365#   3947
SVEC1   022302            4000    4030#
SVEC2   022326            4004    4034#
SVEC3   022344            4008    4037#
SWR     001146             973#   1260    1282#   1284    1290#   1297#   1348    1371    1374    3933    3972    3975    5165
                          5179    5187    5194    5240    5247    5259    5262    5623    5638#
SWREG   000176             907#   1290
SW0    = 000001            834#
SW00   = 000001            824#    834    1374
SW01   = 000002            823#    833    1371    3933
SW02   = 000004            822#    832
SW03   = 000010            821#    831
SW04   = 000020            820#    830
SW05   = 000040            819#    829
SW06   = 000100            818#    828
SW07   = 000200            817#    827
SW08   = 000400            816#    826
SW09   = 001000            815#    825
SW1    = 000002            833#
SW10   = 002000            814#   3972
SW11   = 004000            813#
SW12   = 010000            812#   3975
SW13   = 020000            811#
SW14   = 040000            810#
SW15   = 100000            809#
SW2    = 000004            832#
SW3    = 000010            831#
SW4    = 000020            830#
SW5    = 000040            829#
SW6    = 000100            828#
SW7    = 000200            827#
SW8    = 000400            826#
SW9    = 001000            825#
TBITVE= 000014             867#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
TKVEC = 000060            874#
TPVEC = 000064            875#
TRAPVE= 000034            873#   1268*   1269*
TRTVEC= 000014            868#
TST1    003556           1391#
TST10   006126           1828#
TST11   006416           1889#
TST12   006702           1953#
TST13   007176           2019#
TST14   007424           2071#
TST15   007662           2125#
TST16   010170           2194#
TST17   010516           2267#
TST2    004042           1455#
TST20   010766           2324#
TST21   011256           2385#
TST22   011542           2449#
TST23   012036           2515#
TST24   012264           2567#
TST25   012522           2620#
TST26   013030           2689#
TST27   013356           2762#
TST3    004336           1521#
TST30   013626           2819#
TST31   014116           2880#
TST32   014402           2944#
TST33   014676           3010#
TST34   015162           3074#
TST35   015456           3140#
TST36   015764           3209#
TST37   016312           3282#
TST4    004564           1573#
TST40   016620           3351#
TST41   017146           3424#
TST42   017432           3487#
TST43   017726           3553#
TST44   020234           3621#
TST45   020562           3693#
TST46   C_1232           3798#
TST5    005022           1627#
TST6    005330           1597#
TST7    005656           1771#
TYPE  = 104401           1310    1318    1320    1353    1358    3974    4002    4006    4010    4023    4025    5242    5250
                         5299    5320    5322    5323    5326    5328    5331    5334    5341    5343    5348    5404    5557
                         5605#   5647
TYPOC = 104402           5308    5330    5333    5338    5606#
TYPON = 104404           5608#
TYPOS = 104403           1356    4018    5607#
WFLAG = 000344            884#   1379    3943
XHI     034112           5630    5834    5865#
XLO     034102           5828    5832    5863#
XOP1    034122           5836    5848    5867#
XOP2    034130           5838    5850    5868#
XOP3    034136           5840    5852    5869#
XOP4    034144           5842    5854    5870#
XOP5    034160           5844    5856    5872#
```

```
XOP6    034152          5846    5858    5871#
$ADD    023640          1404    1468    1534    1586    1640    1710    1784    1841    3720    3822    4329#
$APTHD  001000           931     937#
$ASTAT= ****** U        5474    5489
$ATYC   031230          5445    5447#
$ATY1   031204          5443#
$ATY3   031212          5389    5444#
$ATY4   031222          5255    5446#
$AUTOB  001142           970#
$BDRDR  001130           965#
$BDDAT  001134           967#
$BELL   001172           985#   3974    5242    5274
$CHARC  031200          5406#   5416#   5423    5432#   5437#
$CKSWR= ****** U        5611
$CMTAG  001100           952#   1257    1258    1266    1272    1273    1274
$CM3  = 000000           983#
$CONV   027672          4333    4665    4924    5140#
$CPUOP  001230          1008#
$CRLF   001177           987#   1358    5250    5274    5300    5322    5328    5343    5405    5440
$DEVCT  001212           999#
$DIV    026434          2893    2957    3023    3087    3153    3222    3295    3364    3437    3500    3566    3634    3719
                        3824    3829    4920#
$DORGN  022170          3968    3979    3987#
$ENDAD  022160           917    3982#   5269
$ENDCT  022114          1272    1350    3970#
$ENV    001222          1004#   5252    5384    5452    5476
$ENVM   001223          100?#   1295    5386    5391    5454
$EOP    022054          3931    3934    3946    3960#
$EOPCT  022106          1272#   1350    3967#   3971
$ERFLG  001104           955#   3961#   5183    5185    5191#   5213    5237#   5274
$ERMAX  001122           962#   1275#   5185    5208#   5213
$ERROR  030220          1266    5227#
$ERRPC  001124           963#   5244#   5245#   5246    5274    5307    5355
$ERRTB  001232          1019#   5317
$ERTTL  001116           960#   5243#   5274
$ESCAP  001170           984#   1274#   5207#   5265    5267    5274
$ETABL  001222          1003#
$ETEND  001232           943    1015#
$FATAL  001204           996#   5480#
$FEA    002404          1174#   1395#   1433    1459#   1497    1525#   1577#   1631#   1669    1701#   1739    1775#   1832#
                        1893#   1931    1957#   1995    2023#   2075#   2129#   2167    2198#   2236    2271#   2328#   2389#
                        2427    2453#   2491    2519#   2571#   2624#   2662    2693#   2731    2766#   2823#   2884#   2922
                        2948#   2986    3014#   3052    3078#   3116    3144#   3182    3213#   3251    3286#   3324    3355#
                        3393    3428#   3466    3491#   3529    3557#   3595    3625#   3663    3698#   3803#   4485#   4505#
                        4562#   4743#   4754#   4785#   4936#   5030#   5041#   5062#   5814    5822
$FEC    002402          1173#   1394#   1429    1458#   1493    1524#   1576#   1630#   1665    1700#   1735    1774#   1831#
                        1892#   1927    1956#   1991    2022#   2074#   2128#   2163    2197#   2232    2270#   2327#   2388#
                        2423    2452#   2487    2518#   2570#   2623#   2658    2692#   2727    2765#   2822#   2883#   2918
                        2947#   2982    3013#   3048    3077#   3112    3143#   3178    3212#   3247    3285#   3320    3354#
                        3389    3427#   3462    3490#   3525    3556#   3591    3624#   3659    3697#   3802#   4484#   4504#
                        4561#   4742#   4753#   4784#   4935#   5029#   5040#   5061#   5814    5822
$FFLG   031450          5443#   5446#   5474    5483#   5491#
$FILLC  001164           981#   5409    5440
$FILLS  001163           980#   5440
$FPS    002400          1172#   1393#   1407    1420    1425    1457#   1471    1484    1489    1523#   1537    1550    1575#
                        1589    1602    1629#   1643    1656    1661    1699#   1713    1726    1731    1773#   1787    1800
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
                    1830*  1844   1857   1891*  1905   1918   1923   1955*  1969   1982   1987   2021*  2035
                    2048   2073*  2087   2100   2127*  2141   2154   2159   2196*  2210   2223   2228   2269*
                    2283   2296   2326*  2340   2353   2387*  2401   2414   2419   2451*  2465   2478   2483
                    2517*  2531   2544   2569*  2583   2596   2622*  2636   2649   2654   2691*  2705   2718
                    2723   2764*  2778   2791   2821*  2835   2848   2882*  2896   2909   2914   2946*  2960
                    2973   2978   3012*  3026   3039   3044   3076*  3090   3103   3108   3142*  3156   3169
                    3174   3211*  3225   3238   3243   3284*  3298   3311   3316   3353*  3367   3380   3385
                    3426*  3440   3453   3458   3489*  3503   3516   3521   3555*  3569   3582   3587   3623*
                    3637   3650   3655   3696*  3728   3743   3801*  3835   3850   4249   4263   4330*  4331
                    4478*  4480   4483*  4498*  4500   4503*  4520*  4523*  4531   4556   4560*  4662*  4663
                    4738   4741*  4747*  4749   4752*  4778*  4780   4783*  4798*  4801*  4809   4921*  4922
                    4934*  5025   5028*  5034*  5036   5039*  5055*  5057   5060*  5072*  5075*  5083   5813
                    5822
$GDAOR   001126      964#
$GDOAT   001132      966#
$GET42   022150     3976   3978#
$GTSWR=  ******  U  5610
$HD    = 000000     749
$HIBTS   001000      938#
$ICNT    001106      956#  5198*  5199   5201*  5212
$ILLUP   032120     5615   5631   5652#
$INTAG   001143      971#
$ITEMB   001120      961#  5246*  5254   5274   5304
$LF      001200      988#  5274   5440
$LFLG    031447     5484*  5490#
$LPADR   001110      957#  1276*  1413*  1477*  1543*  1595*  1649*  1719*  1793*  1850*  1911*  1975*  2041*
                    2093*  2147*  2216*  2289*  2346*  2407*  2471*  2537*  2589*  2642*  2711*  2784*  2841*
                    2902*  2966*  3032*  3096*  3162*  3231*  3304*  3373*  3446*  3509*  3575*  3643*  3729*
                    3836*  5189*  5205*  5210   5212
$LPERR   001114      959#  1277*  5189   5206*  5212   5264
$LPTST   001112      958#  5181
$MAIL    001202      939    943    994#   1294   5204   5252   5384
$MBADR   001002      939#
$MFLG    031446     5444*  5450   5485*  5489#
$MSGAD   001216     1001#  5460*  5463
$MSGLG   001220     1002#  5465*
$MSGTY   001202      995#  5458   5466*  5478   5482*
$MUL     025314     2398   2462   2528   2580   2633   2702   2775   2832   3721   3723   3830   4661#
$MXCNT   030216     5202   5212#
$NULL    001162      979#  5411   5440
$NWTST=  000001     1388#  1452#  1518#  1570#  1624#  1694#  1768#  1825#  1886#  1950#  2016#  2068#  2122#
                    2191#  2264#  2321#  2382#  2446#  2512#  2564#  2617#  2686#  2759#  2816#  2877#  2941#
                    3007#  3071#  3137#  3206#  3279#  3348#  3421#  3484#  3550#  3618#  3690#  3795#
$OCNT    031674     5529*  5558*  5571#
$OMODE   031676     5524*  5528*  5533   5536*  5547*  5573#
$OVER    030202     5166   5182   5190   5200   5209#
$PASS    001210      998#  1294*  1354   3964*  3965*  5196   5213
$PASTM   001006      941#
$POLSH   023560     1401   1465   1531   1583   1637   1707   1781   1838   1899   1963   2029   2081   2135
                    2204   2277   2334   2395   2459   2525   2577   2630   2699   2772   2829   2890   2954
                    3020   3084   3150   3219   3292   3361   3434   3497   3563   3631   3712   3819   4243#
$POPX    023612     1405   1469   1535   1587   1641   1711   1785   1842   1903   1967   2033   2085   2139
                    2208   2281   2338   2399   2463   2529   2581   2634   2703   2776   2833   2894   2958
                    3024   3088   3154   3223   3296   3365   3438   3501   3567   3635   3724   3831   4260#
$POWER   032126     5648   5655#
$PUSH    023562     1402   1403   1466   1467   1532   1533   1584   1585   1638   1639   1708   1709   1782
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
                        1783    1839    1840    1900    1901    1964    1965    2030    2031    2082    2083    2136    2137
                        2205    2206    2278    2279    2335    2336    2396    2397    2460    2461    2526    2527    2578
                        2579    2631    2632    2700    2701    2773    2774    2830    2831    2891    2892    2955    2956
                        3021    3022    3085    3086    3151    3152    3220    3221    3293    3294    3362    3363    3435
                        3436    3498    3499    3564    3565    3632    3633    3713    3714    3716    3717    3718    3722
                        3820    3821    3823    3825    3826    3828    4248#
$PWRAD  032114          5650#
$PWRDN  031746          1270    5615#   5645
$PWRMG  032110          5648#
$PWRUP  032020          5625    5631#
$QUES   001176           986#   5274    5440
$RDCHR= ****** U        5611
$RDDEC= ****** U        5611
$RDLIN= ****** U        5611
$RDOCT= ****** U        5611
$R2A  = ****** U        5611
$SAVRE= ****** U        5611
$SAVR6  032124          5624#   5632    5633#   5634#   5654#
$SCOPE  027742          1264    5163#
$SETUP= 000037          1248#   1263    1264    1266    1268    1270    1272    1273    1274    1276    5164    5228    5262
                        5269
$STUP = 177777          1248#
$SUB    023634          1902    1966    2032    2084    2138    2207    2280    2337    3715    3827    4328#
$SVLAD  030146          5174    5203#
$SVPC = 000204           915#    920
$SWR  = 167400           738#    749     983     984     985    1273    1274    1276    1277    1392    1456    1522    1574
                        1628    1698    1772    1829    1890    1954    2020    2072    2126    2195    2268    2325    2386
                        2450    2516    2568    2621    2690    2763    2820    2881    2945    3011    3075    3141    3210
                        3283    3352    3425    3488    3554    3622    3694    3799    5155    5156    5157    5158    5159
                        5165    5177    5179    5180    5183    5184    5185    5192    5193    5194    5206    5209    5212
                        5219    5220    5221    5222    5223    5240    5247    5259    5262    5274    5651
$SWREG  001224          1006#   1297
$SWRMK= 000000          5159    5181
$TESTN  001206           997#   5204*   5354
$TIMES  001166           983#   1273*   3963*   5192*   5199    5202*   5212
$TKB    001154           976#
$TKS    001152           975#
$TN   = 000047           738#    749     1388    1392#   1452    1456#   1518    1522#   1570    1574#   1624    1628#   1694
                        1698#   1768    1772#   1825    1829#   1886    1890#   1950    1954#   2016    2020#   2068    2072#
                        2122    2126#   2191    2195#   2264    2268#   2321    2325#   2382    2386#   2446    2450#   2512
                        2516#   2564    2568#   2617    2621#   2686    2690#   2759    2763#   2816    2820#   2877    2881#
                        2941    2945#   3007    3011#   3071    3075#   3137    3141#   3206    3210#   3279    3283#   3348
                        3352#   3421    3425#   3484    3488#   3550    3554#   3618    3622#   3690    3694#   3795    3799#
$TPB    001160           978#   5429*   5440
$TPFLG  001165           982#   5378    5440
$TPS    001156           977#   5427    5440
$TRAP   031700          1268    5582#
$TRAP2  031722          5593#   5604
$TRP  = 000005          5597#   5606#   5607#   5608#   5609#
$TRPAD  031734          5587    5604#
$TSTM   001004           940#
$TSTNM  001102           954#   1387*   3962*   5181    5203*   5204    5209    5213    5239    5274
$TYPBN= ****** U        5609
$TYPDS= ****** U        5609
$TYPE   030722          5378#   5471    5597    5605
$TYPEC  031134          5408    5415    5422    5427#   5428
```

```
$TYPER  030462            5249    5298#
$TYPEX  031202            5433    5435    5438#
$TYPOC  031476            5527#   5606
$TYPON  031512            5526    5529#   5608
$TYPOS  031452            5522#   5607
$UNIT   001214            1000#
$UNITM  001010             942#
$USWR   001226            1007#
$XTSTR  027752            5168#
$OFILL  031675            5523*   5527*   5537    5572#
$40CAT= ****** U          5165    5249
.       = 034166           901#    905#    915     916#    918#    920#    921#    927     928#    930#    932#    951#    989
                          1261    1276    1277    1413    1477    1543    1595    1649    1719    1793    1850    1911    1975
                          2041    2093    2147    2216    2289    2346    2407    2471    2537    2589    2642    2711    2784
                          2841    2902    2966    3032    3096    3162    3231    3304    3373    3446    3509    3575    3643
                          3729    3836    4129#   5212    5213    5274    5360#   5440    5492#   5627    5653    5812#
.$ASTA= ****** U          5444    5447
.$X     = 001000           927#    932
```

```
CMPFLT    738#
COMMEN    877#
COMM00    738#
COMM01    738#
COMM02    738#
COMM03    738#
COMM04    738#
COMM05    738#
COMM06    738#
COMM07    738#
COMM1     738#
COMM10    738#
COMM11    738#
COMM12    738#
COMM13    738#
COMM14    738#
COMM15    738#
COMM16    738#
COMM17    738#
COMM2     738#
COMM20    738#
COMM21    738#
COMM22    738#
COMM23    738#
COMM24    738#
COMM25    738#
COMM26    738#
COMM27    738#
COMM3     738#
COMM30    738#
COMM31    738#
COMM32    738#
COMM33    738#
COMM34    738#
COMM35    738#
COMM36    738#
COMM37    738#
COMM4     738#
COMM40    738#
COMM41    738#
COMM42    738#
COMM43    738#
COMM44    738#
COMM45    738#
COMM46    738#
COMM47    738#
ENDCOM    877#
ERRCMP    738#
ERRLUR    738#  5273
ERROR     771#  1423  1431  1435  1448  1487  1495  1499  1514  1553  1566  1605  1620  1659  1667
       1671     1689  1729  1737  1741  1763  1803  1821  1860  1882  1921  1929  1933  1946  1985
       1993     1997  2012  2051  2064  2103  2118  2157  2165  2169  2187  2226  2234  2238  2260
       2299     2317  2356  2378  2417  2425  2429  2442  2481  2489  2493  2508  2547  2560  2599
       2614     2652  2660  2664  2682  2721  2729  2733  2755  2794  2812  2851  2873  2912  2920
       2924     2937  2976  2984  2988  3003  3042  3050  3054  3067  3106  3114  3118  3133  3172
       3180     3184  3202  3241  3249  3253  3275  3314  3322  3326  3344  3383  3391  3395  3417
```

```
        3456    3464    3468    3481    3519    3527    3531    3546    3585    3593    3597    3615    3653    3661    3665
        3687    3746    3791    3853    3910    4143
ESCAPE   877#
FCOM0    738#
FCOM1    738#
FCOM2    738#
FCOM3    738#
FCOM4    738#
FPRGT0   738#
FPRGT1   738#
FPSFEC   738#
FPSTST   738#
GENCOM   738#
GENTS1   738#
GENTS2   738#
GENTS3   738#
GENTS4   738#
GETPRI   877#
GETSWR   877#
GTSTD    738#
GTSTF    738#
HTSTD    738#
HTSTF    738#
MOVDIS   738#
MULT     877#
NEWTST   877#   1388    1452    1518    1570    1624    1694    1768    1825    1886    1950    2016    2068    2122    2191
        2264    2321    2382    2446    2512    2564    2617    2686    2759    2816    2877    2941    3007    3071    3137
        3206    3279    3348    3421    3484    3550    3618    3690    3795
POP      877#   5486    5487    5638    5639
PUSH     877#   5447    5449    5470    5617    5623
REPORT   877#
SBTST1   738#
SBTST2   738#
SCOM0    738#
SCOM1    738#
SCOM2    738#
SCOM3    738#
SCOM4    738#
SCOM5    738#
SCOM6    738#
SCOM7    738#
SCOPE    772#   1391    1455    1521    1573    1627    1697    1771    1828    1889    1953    2019    2071    2125    2194
        2267    2324    2385    2449    2515    2567    2620    2689    2762    2819    2880    2944    3010    3074    3140
        3209    3282    3351    3424    3487    3553    3621    3693    3798    3917    3960
SCPLUR   738#   5164
SEADAT   738#
SETPRI   877#
SETREG   738#   5228
SETTRA  5597#   5606    5607    5608
SETUP    738#    877#   1255
SKIP     877#
SLASH    877#   1312    1339
SPACE    877#
STARS    877#    913     924     926     933     947     989     992    1251    1253    1342    1344    1361    1363    1389
        1390    1452    1454    1518    1520    1570    1572    1624    1626    1694    1696    1768    1770    1825    1827
        1886    1888    1950    1952    2016    2018    2068    2070    2122    2124    2191    2193    2264    2266    2321
```

```
        2323    2382    2384    2446    2448    2512    2514    2564    2566    2617    2619    2686    2688    2759    2761
        2816    2818    2877    2879    2941    2943    3007    3009    3071    3073    3137    3139    3206    3208    3279
        3281    3348    3350    3421    3423    3484    3486    3550    3552    3618    3620    3690    3692    3731    3795
        3797    3838    3914    3950    5152    5215    5274    5363    5442    5499    5576    5613    5629
STATUS   738#
SWRSU    877#  1278#
TADD01   738#
TADD02   738#
TADDF1   738#
TADDF2   738#
TADDR1   738#
TADDR2   738#
TRMTRP  5597#
TYPBIN   877#
TYPDEC   877#
TYPNAM   877#
TYPNUM   877#
TYPOCS   877#
TYPOCT   877#
TYPTXT   877#
UPCODE   738#  5636
$$CMRE   945#
$$CMTM   945#
$$ESCA   877#
$$NEWT   877#  1388    1452    1518    1570    1624    1694    1768    1825    1886    1950    2016    2068    2122    2191
        2264    2321    2382    2446    2512    2564    2617    2686    2759    2816    2877    2941    3007    3071    3137
        3206    3279    3348    3421    3484    3550    3618    3690    3795
$$SET   5597#  5606    5607    5608
$$SETM  1294#
$$SKIP   877#
.EQUAT   738#   767
.HEADE   738#   739
.SBPAS   738#  3914
.SETUP   738#  1248
.STPAS   738#  1312
.$ACTI   738#   911
.$APTB   738#   990#
.$APTH   738#   922
.$APTY   738#  5440
.$CATC   738#   899
.$CMTA   738#   945
.$EOP    738#
.$ERRO   738#  5213
.$POWE   738#  5611
.$SCOP   738#  5150
.$TRAP   738#  5574
.$TYER   738#
.$TYPE   738#  5361
.$TYPO   738#  5497


. ABS.   034166      000

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

```
 DSKZ:DQFPDA,DSKZ:DQFPDA.SEQ/SOL/CRF=DQFPDA.MEM,DQFPDA.MAC,DQFPDA.P11
 RUN-TIME: 20 18 1 SECONDS
 RUN-TIME RATIO: 126/40=3.1
 CORE USED:  32K  (64 PAGES)
```