

M8188,
FPF11

FPF11 FLT PNT DIAG #1
CJFPAAO

AH-S441A-MC
FICHE 1 OF 1

OCT 1981
COPYRIGHT © 1981
MADE IN USA



The main body of the document is a large grid of approximately 15 columns and 25 rows of small, dense tables. Each cell in the grid contains a small table with multiple columns and rows of text, likely representing diagnostic data or flight parameters. The text is too small to be legible, but the overall structure is a comprehensive data matrix.

IDENTIFICATION

PRODUCT CODE: AC-S439A-MC
PRODUCT NAME: CJFPAA0 FPF11 FLT PNT DIAG #1
DATE CREATED: JANUARY 1981
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: GUS PASQUANTONIO

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1981 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	HARDWARE
2.2	SOFTWARE
3.0	LOADING
4.0	STARTING
5.0	RUN TIME OPTIONS
6.0	ERROR HANDLING
7.0	RESTRICTIONS
8.0	MISCELLANEOUS
8.1	EXECUTION TIME
8.2	ACT, APT AND XXDP COMPATIBILITY
8.3	EXPANSION HOOKS
9.0	TEST DESCRIPTION
10.0	PROGRAM LISTING

1.0

ABSTRACT

=====

THE FPF11 DIAGNOSTIC PACKAGE CONSISTS OF TWO PROGRAMS DESIGNED TO DETECT AND REPORT LOGIC FAULTS IN THE LSI 11/23 FPF11 FLOATING POINT PROCESSOR (M8188).

THROUGH THE USE OF ALL ADDRESSING MODES, AND CAREFUL SELECTION OF OPERANDS, ALL ROM MICRO STATES ARE EXECUTED AND ALL BRANCH MICRO TESTS (BUT'S) ARE TAKEN PROVIDING A THOROUGH VERIFICATION OF THE MICROCODE AND LOGIC.

PART 1 (CJFPAA) TESTS THE FOLLOWING FUNCTIONS:

LDFPS, STFPS, AND CFCC
SETF, SETD, SETI AND SETL
STST (DST MODE 0)
LDF AND LDD (ALL FSRC MODES)
STD (FDST MODE 0 AND 1)
ADDF, ADDD AND SUBD
CMPD AND CMPF
DIVD AND DIVF
MULD AND MULF
MODD AND MODF

AND PART 2 (CJFPBA) FINISHES UP WITH:

STF AND STD (ALL FDST MODES)
STCFD AND STCDF
CLRD AND CLRF
NEGF AND NEGD
ABSF AND ABSD
TSTF AND TSTD
NEGF, ABSF AND TSTF (ALL FDST MODES)
NEGF, ABSF AND TSTF (ALL FDST MODES)
LDFPS (ALL SRC MODES)
LDCIF AND LDCLF
LDCID AND LDCLD
LDEXP
STFPS (ALL DST MODES)
STCFL AND STCFI
STCDL AND STCDI
STEXP
STST (DST MODE 1)

2.0 REQUIREMENTS =====

2.1 HARDWARE =====

LSI 11/23 CPU WITH A MINIMUM OF 16K MEMORY.
CONSOLE TERMINAL.
XXDP (OR XXDP+) SYSTEM DEVICE AND DIAGNOSTIC MEDIA.
ONE (1) M8188 FPF11 MODULE (AND ASSOCIATED CABLE).

2.2 SOFTWARE =====

THESE PROGRAMS ASSUME THAT THE 11/23 CPU IS FULLY OPERATIONAL.
IF NOT (OR YOU DON'T KNOW), RUN ALL APPLICABLE 11/23 CPU
DIAGNOSTICS.

3.0 LOADING =====

THESE PROGRAMS ARE PROVIDED ON AN XXDP (XXDP+) MEDIA.
USE STANDARD XXDP (XXDP+) LOADING PROCEDURES.

4.0 STARTING =====

BOTH PROGRAMS START/RESTART AT ADDRESS 200.
THE SWITCH REGISTER IS DISPLAYED AT START TIME, AND
MAY BE ALTERED AS REQUIRED (SEE BELOW FOR SWR OPTIONS).

5.0 RUN TIME OPTIONS =====

THE FOLLOWING SWITCH REGISTER OPTIONS ARE SUPPORTED:

SWITCH -----	OCTAL -----	FUNCTION -----
15	100000	HALT ON ERROR
14	040000	LOOP ON CURRENT TEST
13	020000	INHIBIT ERROR PRINTOUT
12	010000	PRINT TEST NUMBERS
11	004000	INHIBIT ITERATIONS (NA)
10	002000	RING BELL ON ERROR
09	001000	LOOP ON ERROR
08	000400	LOOP ON TEST IN SWR<6:0>
07	000200	RESERVED FOR LOGIC ANALYZER
06:00	0000NN	TEST NUMBER (FOR SWR<8>)

THE SWITCH REGISTER MAY BE ALTERED ON-THE-FLY BY TYPING
CONTROL G <^G>. THE PROGRAM WILL DISPLAY THE CURRENT
SWITCH SETTINGS AND WAIT FOR A NEW ONE. IF NO CHANGE,
JUST TYPE A <CR> TO CONTINUE ON USING THE CURRENT SWITCHES.

6.0 ERROR HANDLING

=====

ALL ERRORS ARE REPORTED ON THE CONSOLE DEVICE AS THEY OCCUR (UNLESS INHIBITED), FOLLOWED BY RECOVERY ACTION AS DICTATED BY THE CURRENT SWITCH REGISTER SETTING.

NOTE THAT ERROR ANALYSIS ASSUMES A SINGLE FAULT CONDITION IN THE FPF11. MULTIPLE FAULTS MAY CAUSE MISLEADING ERROR SIGNATURES.

7.0 RESTRICTIONS

=====

NONE

8.0 MISCELLANEOUS

=====

8.1 EXECUTION TIME

=====

THE FOLLOWING ARE TYPICAL EXECUTION TIMES:

PART 1 -- 5 SECONDS PER PASS.

PART 2 -- 2 SECONDS PER PASS.

8.2 ACT, APT, XXDP, AND XXDP+ COMPATABILITY

=====

THESE PROGRAMS WERE ASSEMBLED USING THE TRADITIONAL SYSMAC MACRO PACKAGE AND AS SUCH, ARE FULLY COMPATABLE WITH ACT AND APT, AND ARE CHAINABLE UNDER THE XXDP MONITOR.

THEY ARE ALSO COMPATABLE WITH THE XXDP+ MONITOR, BUT NOT (REPEAT -- NOT) WITH THE XXDP+/APT SYSTEM.

8.3 EXPANSION HOOKS

=====

POSSIBLE FUTURE EXPANSION MAY INVOLVE THE ADDITION OF SOME CODE FOR SET-UP AND INITIALIZATION OF A PROGRAMMABLE LOGIC ANALYZER. A BLOCK OF MEMORY AND SWITCH REGISTER BIT 7 ARE RESERVED FOR THIS PURPOSE.

9.0 TEST DESCRIPTION (PART 1).

=====

9.1 TEST 1 -- LDFPS, STFPS, (SRC/DST MODE 0), AND DATA PATHS.

=====

THIS IS A TEST OF THE LDFPS (LOAD FP STATUS) AND STFPS (STORE FP STATUS) INSTRUCTIONS. A COUNT PATTERN IS GENERATED AND RUN THROUGH THE FLOATING POINT STATUS REGISTER. THIS WILL TEST THE 16-BIT TRI STATE BUS WHICH CONNECTS THE CPU WITH THE FPP AND ALSO RUNS INTERNALLY WITHIN THE FPP. SRC/DST MODE 0 IS USED. NOTE THAT BITS 12 AND 13 ARE UNUSED IN FPS AND ARE MASKED OFF.

TO PREVENT LOCKING OUT THE COMPLETION OF THE TEST BECAUSE OF AN EXCESSIVE NUMBER OF ERRORS, ONLY THE FIRST FIVE ERRORS WILL BE REPORTED THEN THE TEST WILL BE COMPLETED AND AN ERROR SUMMARY GIVEN, AS FOLLOWS:

A DYNAMIC RECORD OF THE LOGICAL 'AND' AND 'OR' OF THE FAILING DATA PATTERNS IS KEPT. THESE CAN BE VERY USEFUL IN DETERMINING STUCK BITS. WHEN THE TEST COMPLETES, THIS SUMMARY IS PRINTED IF ANY ERRORS WERE ENCOUNTERED.

9.2 TEST 2 -- COPY CONDITION CODES (CFCC).

=====

THIS IS A TEST OF THE COPY CONDITION CODES INSTRUCTION. INSURE THAT ONLY PSW<3:0> ARE AFFECTED BY THE CFCC, AND THAT THE FP STATUS REGISTER IS UNALTERED. NOTE: THE CPU IS IN 'USER MODE' DURING THIS TEST.

9.3 TEST 3 -- MODE SETUP (SETF, SETD, SETI, AND SETL).

=====

THIS IS A TEST OF THE SETF, SETD, SETI AND SETL INSTRUCTIONS. EACH INSTRUCTION IS EXECUTED WITH THE FPS CLEAR, AND THEN AGAIN WITH THE FPS CONTAINING ALL 1'S (147777). THE RESULTING FPS IS CHECKED IN EACH CASE.

9.4 TEST 4 -- ILLEGAL FP OP-CODE TRAP AND STST (DST MODE 0).

=====

THIS IS A TEST OF THE ILLEGAL FP OP-CODE TRAP FUNCTION. FP OPCODES 170003 THRU 170010 AND 170013 THRU 170077 ARE ILLEGAL (INVALID) AND SHOULD CAUSE A TRAP TO 244 (WHEN INTERRUPTS ARE ENABLED). WHEN ANY OF THE ABOVE OPCODES TRAP, THE FEC REGISTER SHOULD BE SET TO CODE 2, TO INDICATE AN OPCODE ERROR.

- 9.5 TEST 5 -- FP INTERRUPT DISABLE BIT (FID).
 =====
 THIS IS A TEST OF THE INTERRUPT DISABLE (FID) BIT.
 AN ILLEGAL INSTRUCTION IS EXECUTED WITH FID = 1.
 NO INTERRUPT SHOULD OCCUR.
- 9.6 TEST 6 -- LDD AND STD (FSRC MODE 1).
 =====
 THIS IS A TEST OF THE LOAD AND STORE DOUBLE INSTRUCTIONS.
 LDD (R0),ACO
 STD (ACO,(R0)
- 9.7 TEST 7 -- LDD AND LDF (FSRC MODE 0).
 =====
 THIS IS A TEST OF THE FSRC MODE 0, USING THE INSTRUCTIONS
 LDD AC1,ACO
 LDF AC1,ACO
- 9.10 TEST 10 -- STD AND STF (FDST MODE 0).
 =====
 THIS IS A TEST OF THE STORE INSTRUCTIONS, STD AND STF.
- 9.11 TEST 11 -- ACCUMULATORS WITH FLOATING DATA PATTERNS.
 =====
 THIS IS A TEST OF THE FLOATING POINT PROCESSOR ACCUMULATORS.
 EACH ACCUMULATOR IS TESTED IN TWO WAYS:
 1. A TEST PATTERN GENERATED BY FLOATING A ONE ACROSS
 A FIELD OF ZEROES.
 2. A TEST PATTERN GENERATED BY FLOATING A ZERO ACROSS
 A FIELD OF ONES.
 EACH OF ACCUMULATORS ACO THROUGH AC5 IS TESTED.
 NOTE THAT ACO IS USED TO ACCESS AC4 AND AC5.
- 9.12 TEST 12 -- ACCUMULATORS DUAL ADDRESSING.
 =====
 THIS TEST PERFORMS A DUAL ADDRESSING TEST ON THE ACCUMULATORS.
 NOTE THAT ACCUMULATOR ZERO IS USED TO ACCESS ALL THE OTHERS.
- 9.13 TEST 13 -- LDD AND ADD (FSRC MODE 0) USING AN ILLEGAL AC.
 =====
 THIS IS A TEST OF FSRC MODE 0 WITH ACCUMULATORS 6 AND 7.
 ANY ATTEMPT TO ACCESS EITHER AC6 OR AC7 SHOULD CAUSE A TRAP
 TO VECTOR 244, WITH FEC = 2 (ILLEGAL FPP INSTRUCTION).

- 9.14 TEST 14 -- LDD (FSRC MODE 2) AUTO-INCREMENT.
=====
- THIS IS A TEST OF FSRC MODE 2, AUTO-INCREMENT MODE.
- 9.15 TEST 15 -- LDD (FSRC MODE 4) AUTO-DECREMENT.
=====
- THIS IS A TEST OF FSRC MODE 4, AUTO-DECREMENT MODE.
- 9.16 TEST 16 -- LDF (FSRC MODE 2) AUTO-INCREMENT.
=====
- THIS IS A TEST OF FSRC MODE 2 (AUTO-INCREMENT) IN SINGLE PRECISION MODE (FD=0).
- 9.17 TEST 17 -- LDD (FSRC MODE 2 WITH R7) PC IMMEDIATE.
=====
- THIS IS A TEST OF FSRC MODE 2 USING GR7 (PC). THIS IS THE "IMMEDIATE" MODE.
- 9.20 TEST 20 -- LDD (FSRC MODE 3) AUTO-INCREMENT DEFERRED.
=====
- THIS IS A TEST OF FSRC MODE 3, AUTO INCREMENT DEFERRED.
- 9.21 TEST 21 -- LDD (FSRC MODE 5) AUTO-DECREMENT DEFERRED.
=====
- THIS IS A TEST OF FSRC MODE 5, AUTO DECREMENT DEFERRED.
- 9.22 TEST 22 -- LDD (FSRC MODE 6) INDEXED.
=====
- THIS IS A TEST OF FSRC MODE 6, INDEX MODE
- 9.23 TEST 23 -- LDD (FSRC MODE 7) INDEX DEFERRED.
=====
- THIS IS A TEST OF FSRC MODE 7, INDEX DEFERRED.
- 9.24 TEST 24 -- LDD AND TEST THE FZ, FN, AND FIUV BUTS.
=====
- THIS IS A TEST OF THE (BUT EZBT Y8) FORK, THE (BUT ENBT) FORK AND (BUT FIUV) FORK IN THE LOAD INSTRUCTION FLOWS. EACH OF THE PATTERNS 0, +NUM, -NUM AND -0 IS LOADED TWICE FIRST WITH INITIAL AC <> 0 AND AGAIN WITH AC = 0. THE FPS IS VERIFIED AFTER EACH LOAD FUNCTION.

- 9.25 TEST 25 -- ADDD/SUBD (FSRC MODE 1), BOTH OPERANDS = 0.
 =====
 THIS IS A TEST OF ADD AND SUB WITH BOTH OPERANDS = 0.
 $0 + 0 = 0$ (HOPEFULLY)
- 9.26 TEST 26 -- ADDD/SUBD (FSRC MODE 1), FSRC OPERAND = 0.
 =====
 THIS IS A TEST OF ADDD AND SUBD WITH JUST THE SRC = 0.
 $N + 0 = N$
 $N - 0 = N$
- 9.27 TEST 27 -- SUBD (FSRC MODE 1), FDST OPERAND = 0.
 =====
 THIS IS A TEST OF THE SUBD WITH THE DST OPERAND = 0.
 BOTH POSITIVE AND NEGATIVE SRC OPERANDS ARE TRIED.
 $0 - (+N) = -N$
 $0 - (-N) = +N$
- 9.30 TEST 30 -- ADDD (FSRC MODE 1), FDST OPERAND = 0.
 =====
 THIS IS A TEST OF THE ADDD WITH THE DST OPERAND = 0.
 BOTH POSITIVE AND NEGATIVE SRC OPERANDS ARE TRIED.
 $0 + (+N) = +N$
 $0 + (-N) = -N$
- 9.31 TEST 31 -- ADDD/ADDF, EQUAL EXPONENTS, AND ROUND/TRUNK (BUT FT).
 =====
 THIS IS A TEST OF THE ADD INSTRUCTIONS WITH THE OPERANDS
 HAVING EQUAL EXPONENTS (ALIGNMENT NOT REQUIRED).
 THE ROUND/TRUNCATE FLOW (BUT FT) IS ALSO TESTED HERE.
- 9.32 TEST 32 -- ADDD/ADDF, ALIGN WITH E(AC) LESS THEN E(FSRC).
 =====
 THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS
 AND THE ALIGN FDST EXPONENT ALGORITHM.
 THE ALIGNMENT CONSTANTS 25 FOR FLOATING, AND 57 FOR DOUBLE
 ARE VERIFIED.
 NOTE THAT E(AC) IS LESS THAN E(SRC) SO THE OPERAND IN AC
 (FDST) IS THE ONE TO BE ALIGNED.

- 9.33 TEST 33 -- ADDD/ADDF, ALIGN WITH E(AC) GREATER THAN E(FSRC).
 =====
 THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS
 AND THE ALIGN FSRC EXPONENT ALGORITHM.
 THIS IS THE SAME AS PRECEDING TEST EXCEPT THAT
 E(AC) IS GREATER THAN E(FSRC) SO THE FSRC OPERAND
 IS THE ONE TO BE ALIGNED.
- 9.34 TEST 34 -- ADDD USING POSITIVE AND NEGATIVE OPERANDS.
 =====
 THIS IS A TEST OF THE ADDD INSTRUCTION USING BOTH
 POSITIVE AND NEGATIVE OPERANDS.
- 9.35 TEST 35 -- SUBD USING POSITIVE AND NEGATIVE OPERANDS.
 =====
 THIS IS A TEST OF THE SUBD INSTRUCTION USING VARIOUS OPERANDS.
- 9.36 TEST 36 -- ADDD/SUBD AND CHECK FINAL NORMALIZATION.
 =====
 TEST THAT THE RESULTS OF AN ADD OR SUBTRACT ARE
 PROPERLY NORMALIZED IN ACO.
 DATA IS SELECTED TO TEST TWO CASES.
 FIRST THE MINIMUM SITUATION REQUIRING ONE
 LEFT SHIFT AND THEN THE MAXIMUM SITUATION
 REQUIRING 56 SHIFTS.
- 9.37 TEST 37 -- ADDD AND CHECK ROUND AND TRUNCATE.
 =====
 THIS IS A TEST OF THE ROUND AND TRUNCATE FLOWS.
 IN PARTICULAR TWO THINGS ARE TESTED:
 FIRST A CONDITION IN WHICH ROUNDING RESULTS
 IN THE NEED FOR RENORMALIZATION, AND
 SECOND, THAT THE FPS CONDITION CODES (N AND Z BITS)
 ARE CORRECTLY SET IN VARIOUS COMBINATIONS.
- 9.40 TEST 40 -- ADDD AND CHECK OVERFLOW AND UNDERFLOW.
 =====
 TEST THAT THE OVERFLOW/UNDERFLOW ALGORITHM CAN CORRECTLY
 DETECT THOSE CONDITIONS AND TAKE THE APPROPRIATE ACTION.
 EACH CONDITION IS TESTED TWICE, ONCE WITH TRAPS
 ENABLED (FIU AND FIV = 1), AND AGAIN WITH TRAPS
 DISABLED (FIU AND FIV = 0).
 WHEN TRAPS ARE ENABLED, THE RETURNED FEC IS ALSO TESTED.

- 9.41 TEST 41 -- LDCFD AND LDCDF (FSRC MODE 2).
=====
- TESTS THE LOAD CONVERTED FUNCTIONS.
- 9.42 TEST 42 -- CMPD (FSRC MODE 1).
=====
- TEST THE CMPD INSTRUCTION WITH A VARIETY OF OPERANDS.
A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE
INSTRUCTION, AND CHECK THE RESULTS.
NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE
OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE
OFFENDING INSTRUCTION ITSELF.
- 9.43 TEST 43 -- DIVD (FSRC MODE 1), WITH DIVISOR = 0.

- THIS IS A TEST OF THE DIVD INSTRUCTION WITH A
ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH
TRAP ENABLED AND TRAPS DISABLED.
- 9.44 TEST 44 -- DIVF (FSRC MODE 1).

- TEST THE DIVF INSTRUCTION WITH A VARIETY OF OPERANDS.
A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE
INSTRUCTION, AND CHECK THE RESULTS.
NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE
OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE
OFFENDING INSTRUCTION ITSELF.
- 9.45 TEST 45 -- DIVD (FSRC MODE 1).
=====
- TEST THE DIVD INSTRUCTION WITH A VARIETY OF OPERANDS.
A COMMON SUBROUTINE IS USED AS DESCRIBED ABOVE.
- 9.46 TEST 46 -- MULF (FSRC MODE 1).
=====
- TEST THE MULF INSTRUCTION WITH A VARIETY OF OPERANDS.
A COMMON SUBROUTINE IS USED AS DESCRIBED ABOVE.
- 9.47 TEST 47 -- MULD (FSRC MODE 1).
=====
- TEST THE MULD INSTRUCTION WITH A VARIETY OF OPERANDS.
A COMMON SUBROUTINE IS USED AS DESCRIBED ABOVE.

9.50 TEST 50 -- MODF (FSRC MODE 1).
=====

THIS IS A TEST OF THE MODF INSTRUCTION.
A COMMON SUBROUTINE IS USED AS DESCRIBED ABOVE.

9.51 TEST 51 -- MODD (FSRC MODE 1).
=====

THIS IS A TEST OF THE MODD INSTRUCTION.
A COMMON SUBROUTINE IS USED AS DESCRIBED ABOVE.

10.0 PROGRAM LISTING

=====

THE PROGRAM LISTING FOLLOWS:

79	BASIC DEFINITIONS
100	TRAP CATCHER
(1)	STARTING ADDRESS(ES)
104	OPERATIONAL SWITCH SETTINGS
108	ACT11 HOOKS
110	APT PARAMETER BLOCK
112	COMMON TAGS
(2)	APT MAILBOX-ETABLE
(1)	ERROR POINTER TABLE
128	INITIALIZE THE COMMON TAGS
139	GET VALUE FOR SOFTWARE SWITCH REGISTER
149	COMMON SUBROUTINES
208	
209	FPF PART 1 TESTS
210	
232	T1 LDFPS, STFPS, AND DATA PATHS -- SRC AND DST MODE 0
286	T2 COPY CONDITION CODES
326	T3 SETF, SETD, SETI, AND SETL
392	T4 ILLEGAL FPP OP-CODE TRAP, AND STST -- DST MODE 0
445	T5 INTERRUPT DISABLE (FID) BIT
492	T6 LDD AND STD -- FSRC AND FDST MODE 1
688	T7 LDD AND LDF -- FSRC MODE 0
837	T10 STD AND STF -- FDST MODE 0
982	T11 ACCUMULATORS -- DATA PATTERNS
1078	T12 ACCUMULATORS -- DUAL ADDRESSING
1216	T13 LDD AND ADDD -- FSRC MODE 0, WITH ILLEGAL ACCUMULATOR
1299	T14 LDD -- FSRC MODE 2 (AUTO-INCREMENT)
1356	T15 LDD -- FSRC MODE 4 (AUTO-DECREMENT)
1413	T16 LDF -- FSRC MODE 2 (AUTO-INCREMENT)
1469	T17 LDD -- FSRC MODE 2 WITH GR7 (PC IMMEDIATE)
1529	T20 LDD -- FSRC MODE 3 (AUTO-INCR DEFFERRED)
1624	T21 LDD -- FSRC MODE 5 (AUTO-DECR DEFFERRED)
1720	T22 LDD -- FSRC MODE 6 (INDEX)
1805	T23 LDD -- FSRC MODE 7 (INDEX DEFFERRED)
1903	T24 TEST FZ, FN, AND FIUV BITS USING LDD
1996	T25 ADD AND SUBD -- FSRC MODE 1, WITH BOTH OPERANDS = 0
2076	T26 ADDD AND SUBD -- FSRC MODE 1, WITH SRC OPERAND = 0
2161	T27 SUBD -- FSRC MODE 1, WITH DST OPERAND = 0
2210	T30 ADDD -- FSRC MODE 1, WITH DST OPERAND = 0
2257	T31 ADDD AND ADDF (EPOXNENTS EQUAL), AND ROUND\TRRUNK (BUT FT)
2369	T32 ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC)
2529	T33 ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC)
2716	T34 ADDD WITH POSITIVE AND NEGATIVE OPERANDS
2873	T35 SUBD WITH POSITIVE AND NEGATIVE OPERANDS
2974	T36 TEST NORMALIZE FLOW USING ADDD AND SUBD
3085	T37 TEST ROUND\TRUNCATE USING ADDD
3245	T40 TEST OVERFLOW\UNDERFLOW USING ADDD
3408	T41 LDCFD AND LDCDF -- FSRC MODE 2
3555	T42 CMPD -- FSRC MODE 1
3729	T43 DIVD -- FSRC MODE 1, WITH DIVISOR - 0
3845	T44 DIVF -- FSRC MODE 1
4016	T45 DIVD -- FSRC MODE 1
4168	T46 MULF -- FSRC MODE 1
4353	T47 MULD -- FSRC MODE 1
4485	T50 MODF -- FSRC MODE 1
4688	T51 MODD -- FSRC MODE 1

4929	
4930	END OF PASS ROUTINE
4932	SCOPE HANDLER ROUTINE
4962	TYPE ROUTINE
4975	GET VALUE FOR SOFTWARE SWITCH REGISTER
4979	BINARY TO OCTAL (ASCII) AND TYPE
4981	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4993	APT COMMUNICATIONS ROUTINE
4995	TTY INPUT ROUTINE
4997	TRAP DECODER
(3)	TRAP TABLE
5025	POWER DOWN AND UP ROUTINES
5027	ERROR HANDLER ROUTINE
5029	ERROR TYPE OUT ROUTINE
5132	ASCII TEXT AND ERROR MESSAGES.
5739	AREA RESERVED FOR LOGIC ANALYZER SET-UP CODE (NOT INCLUDED).

1
2
3
4
5
10
11
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
12
13
14
15
16
17
18
19
20
21
22
23
24
28
37
45
57
67
77
78
79
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)

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

.ENABL ABS,AMA

.TITLE CJFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
;*COPYRIGHT (C) NOV 1980
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY DIAG ENGINEERING
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC- 1-DZQAC-C4), 31 JULY 1980.
;*
$TN=1
$SWR=160000 ;:HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
;*
;* THESE PROGRAMS WERE ADAPTED FROM THE ORIGINAL 11/34 FP11-A
;* DIAGNOSTICS WRITTEN BY ANTHONY S. VEZZA IN SEPTEMBER '76.
;*
;* FPF PART 1 INCLUDES ALL TESTS FOUND IN FP11-A, PARTS 1 AND 2.
;* FPF PART 2 INCLUDES ALL TESTS FOUND IN FP11-A, PART 3.
;* ERROR ANALYSIS AND REPORTING HAS BEEN REWORKED TO ACCOMODATE
;* THE 11/23 FPF11 MICROCODE AND MAINTENANCE PHILOSOPHY.
;*
;:;G.P. JAN '80
;*
PRGSIZ= ^H<LASTAD> ; PROGRAM SIZE IN 1/8 K UNITS (OCTAL).
:*****

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

;*MISCELLANEOUS DEFINITIONS
HT= 11 ;:CODE FOR HORIZONTAL TAB
LF= 12 ;:CODE FOR LINE FEED
CR= 15 ;:CODE FOR CARRIAGE RETURN
CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;:PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;:STACK LIMIT REGISTER
PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;:HARDWARE SWITCH REGISTER
DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
```



```

(1)
(1)
(1) 000000 R0= %0 ;;GENERAL REGISTER
(1) 000001 R1= %1 ;;GENERAL REGISTER
(1) 000002 R2= %2 ;;GENERAL REGISTER
(1) 000003 R3= %3 ;;GENERAL REGISTER
(1) 000004 R4= %4 ;;GENERAL REGISTER
(1) 000005 R5= %5 ;;GENERAL REGISTER
(1) 000006 R6= %6 ;;GENERAL REGISTER
(1) 000007 R7= %7 ;;GENERAL REGISTER
(1) 000006 SP= %6 ;;STACK POINTER
(1) 000007 PC= %7 ;;PROGRAM COUNTER
(1)
(1)
(1) 000000 ;;*PRIORITY LEVEL DEFINITIONS
(1) 000040 PR0= 0 ;;PRIORITY LEVEL 0
(1) 000100 PR1= 40 ;;PRIORITY LEVEL 1
(1) 000140 PR2= 100 ;;PRIORITY LEVEL 2
(1) 000200 PR3= 140 ;;PRIORITY LEVEL 3
(1) 000240 PR4= 200 ;;PRIORITY LEVEL 4
(1) 000300 PR5= 240 ;;PRIORITY LEVEL 5
(1) 000340 PR6= 300 ;;PRIORITY LEVEL 6
(1) 000340 PR7= 340 ;;PRIORITY LEVEL 7
(1)
(1)
(1) 100000 ;;*'SWITCH REGISTER'' SWITCH DEFINITIONS
(1) 040000 SW15= 100000
(1) 020000 SW14= 40000
(1) 010000 SW13= 20000
(1) 004000 SW12= 10000
(1) 002000 SW11= 4000
(1) 001000 SW10= 2000
(1) 000400 SW09= 1000
(1) 000200 SW08= 400
(1) 000100 SW07= 200
(1) 000040 SW06= 100
(1) 000020 SW05= 40
(1) 000010 SW04= 20
(1) 000004 SW03= 10
(1) 000002 SW02= 4
(1) 000001 SW01= 2
(1) SW00= 1
(1) .EQUIV SW09,SW9
(1) .EQUIV SW08,SW8
(1) .EQUIV SW07,SW7
(1) .EQUIV SW06,SW6
(1) .EQUIV SW05,SW5
(1) .EQUIV SW04,SW4
(1) .EQUIV SW03,SW3
(1) .EQUIV SW02,SW2
(1) .EQUIV SW01,SW1
(1) .EQUIV SW00,SW0
(1)
(1)
(1) 100000 ;;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1) 040000 BIT15= 100000
(1) 020000 BIT14= 40000
(1) 010000 BIT13= 20000
(1) BIT12= 10000
    
```

```

(1)      004000      BIT11= 4000
(1)      002000      BIT10= 2000
(1)      001000      BIT09= 1000
(1)      000400      BIT08= 400
(1)      000200      BIT07= 200
(1)      000100      BIT06= 100
(1)      000040      BIT05= 40
(1)      000020      BIT04= 20
(1)      000010      BIT03= 10
(1)      000004      BIT02= 4
(1)      000002      BIT01= 2
(1)      000001      BIT00= 1
(1)      .EQUIV      BIT09,BIT9
(1)      .EQUIV      BIT08,BIT8
(1)      .EQUIV      BIT07,BIT7
(1)      .EQUIV      BIT06,BIT6
(1)      .EQUIV      BIT05,BIT5
(1)      .EQUIV      BIT04,BIT4
(1)      .EQUIV      BIT03,BIT3
(1)      .EQUIV      BIT02,BIT2
(1)      .EQUIV      BIT01,BIT1
(1)      .EQUIV      BIT00,BIT0
(1)
(1)      : *BASIC "CPU" TRAP VECTOR ADDRESSES
(1)      000004      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
(1)      000010      RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
(1)      000014      TBITVEC=14         ;; 'T' BIT
(1)      000014      TRTVEC= 14         ;; TRACE TRAP
(1)      000014      BPTVEC= 14         ;; BREAKPOINT TRAP (BPT)
(1)      000020      IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)      000024      PWRVEC= 24         ;; POWER FAIL
(1)      000030      EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
(1)      000034      TRAPVEC=34        ;; 'TRAP' TRAP
(1)      000060      TKVEC= 60          ;; TTY KEYBOARD VECTOR
(1)      000064      TPVEC= 64          ;; TTY PRINTER VECTOR
(1)      000240      PIRQVEC=240        ;; PROGRAM INTERRUPT REQUEST VECTOR
(1)
(1)      :
(1)      : FPF-11 DEFINITIONS.
(1)      :
(1)      80
(1)      81
(1)      82
(1)      83      000244      FPVEC= 244          ; THE STANDARD FP VECTOR...
(1)      84      000000      AC0= %0          ; ...AND ACCUMULATORS.
(1)      85      000001      AC1= %1
(1)      86      000002      AC2= %2
(1)      87      000003      AC3= %3
(1)      88      000004      AC4= %4
(1)      89      000005      AC5= %5
(1)      90      000006      AC6= %6
(1)      91      000007      AC7= %7
(1)      92
(1)      93      : MISCELLANEOUS DEFINITIONS.
(1)      94
(1)      95      000011      TAB= HT
(1)      96      000401      SKP1= BR+1
(1)      97      000402      SKP2= BR+2
(1)      98      000403      SKP3= BR+3
(1)      99
    
```

```

100      .SBTTL TRAP CATCHER
(1)
(1)      000000      .=0
(1)      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A '+2,HALT'
(1)      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)      000174      .=174
(1) 000174 000000  DISPRFG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
(1) 000176 000000  SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
(1)      .SBTTL STARTING ADDRESS(ES)
(1) 000200 000137 003542 JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

101
102      167400      $SWR= 167400      ; RE-DEFINE SWITCH 12.
103      000200      $SWRMK- 000200      ; RE-DEFINE TEST NUMBER FIELD.
104      .SBTTL OPERATIONAL SWITCH SETTINGS
(1)      ;*
(1)      ;* SWITCH USE
(1)      ;* -----
(1)      ;* 15 HALT ON ERROR
(1)      ;* 14 LOOP ON TEST
(1)      ;* 13 INHIBIT ERROR TYPEOUTS
(1)      ;* 12 PRINT TEST NUMBERS
(1)      ;* 11 INHIBIT ITERATIONS
(1)      ;* 10 BELL ON ERROR
(1)      ;* 9 LOOP ON ERROR
(1)      ;* 8 LOOP ON TEST IN SWR<6:0>
(1)      ;* 7 ENABLE ANALYZER (RESERVED)
105
106
107      001000      .=1000
108      .SBTTL ACT11 HOOKS
(1)
(2)      ;*****
(1)      ;HOOKS REQUIRED BY ACT11
(1)      001000      $SVPC=.      ;SAVE PC
(1)      000046      .=46
(1) 000046 037464  $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)      000052      .=52
(1) 000052 000000  .WORD 0      ;;2)SET LOC.52 TO ZERO
(1)      001000      .= $SVPC      ;; RESTORE PC
109
110      .SBTTL APT PARAMETER BLOCK
(1)
(2)      ;*****
(1)      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)      ;*****
(1)      001000      .$X=.      ;;SAVE CURRENT LOCATION
(1)      000024      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1) 000024 000200  200      ;;FOR APT START UP
(1)      000044      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1) 000044 001000  $APTHDR ;;POINT TO APT HEADER BLOCK
(1)      001000      .=.$X      ;;RESET LOCATION COUNTER
(2)      ;*****
(1)      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)      ;INTERFACE SPEC.
(1)
(1) 001000  $APTHD:
    
```

(1)	001000	000000	\$HIBTS: .WORD	0	:: TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1)	001002	001234	\$MBADR: .WORD	\$MAIL	:: ADDRESS OF APT MAILBOX (BITS 0-15)
(1)	001004	000005	\$TSTM: .WORD	5	:: RUN TIM OF LONGEST TEST
(1)	001006	000010	\$PASTM: .WORD	10	:: RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1)	001010	000000	\$UNITM: .WORD	0	:: ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1)	001012	000027	.WORD	\$ETEND-\$MAIL/2	:: LENGTH MAILBOX-ETABLE (WORDS)
111					

```

112      .SBTTL COMMON TAGS
(1)
(2)      ::*****
(1)      ::THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)      ::USED IN THE PROGRAM.
(1)
(1)      001100      .=1100
(1)      001100      000000      $CMTAG:      .WORD      0      ;;START OF COMMON TAGS
(1)      001102      000      $TSTNM: .BYTE      0      ;;CONTAINS THE TEST NUMBER
(1)      001103      000      $ERFLG: .BYTE      0      ;;CONTAINS ERROR FLAG
(1)      001104      000000      $ICNT:  .WORD      0      ;;CONTAINS SUBTEST ITERATION COUNT
(1)      001106      000000      $LPADR: .WORD      0      ;;CONTAINS SCOPE LOOP ADDRESS
(1)      001110      000000      $LPERR: .WORD      0      ;;CONTAINS SCOPE RETURN FOR ERRORS
(1)      001112      000000      $ERTTL: .WORD      0      ;;CONTAINS TOTAL ERRORS DETECTED
(1)      001114      000      $ITEMB: .BYTE      0      ;;CONTAINS ITEM CONTROL BYTE
(1)      001115      001      $ERMAX: .BYTE      1      ;;CONTAINS MAX. ERRORS PER TEST
(1)      001116      000000      $ERRPC: .WORD      0      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)      001120      000000      $GDADR: .WORD      0      ;;CONTAINS ADDRESS OF 'GOOD' DATA
(1)      001122      000000      $BDADR: .WORD      0      ;;CONTAINS ADDRESS OF 'BAD' DATA
(1)      001124      000000      $GDDAT: .WORD      0      ;;CONTAINS 'GOOD' DATA
(1)      001126      000000      $BDDAT: .WORD      0      ;;CONTAINS 'BAD' DATA
(1)      001130      000000      .WORD      0      ;;RESERVED--NOT TO BE USED
(1)      001132      000000      .WORD      0
(1)      001134      000      $AUTOB: .BYTE      0      ;;AUTOMATIC MODE INDICATOR
(1)      001135      000      $INTAG: .BYTE      0      ;;INTERRUPT MODE INDICATOR
(1)      001136      000000      .WORD      0
(1)      001140      177570      SWR:      .WORD      DSWR      ;;ADDRESS OF SWITCH REGISTER
(1)      001142      177570      DISPLAY: .WORD      DDISP      ;;ADDRESS OF DISPLAY REGISTER
(1)      001144      177560      $TKS:      177560      ;;TTY KBD STATUS
(1)      001146      177562      $TKB:      177562      ;;TTY KBD BUFFER
(1)      001150      177564      $TPS:      177564      ;;TTY PRINTER STATUS REG. ADDRESS
(1)      001152      177566      $TPB:      177566      ;;TTY PRINTER BUFFER REG. ADDRESS
(1)      001154      000      $NULL:  .BYTE      0      ;;CONTAINS NULL CHARACTER FOR FILLS
(1)      001155      002      $FILLS: .BYTE      2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)      001156      012      $FILLC: .BYTE      12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
(1)      001157      000      $TPFLG: .BYTE      0      ;;'TERMINAL AVAILABLE' FLAG (BIT<07>-0 YES)
(3)      001160      000000      $TMP0:  .WORD      0      ;;USER DEFINED
(3)      001162      000000      $TMP1:  .WORD      0      ;;USER DEFINED
(3)      001164      000000      $TMP2:  .WORD      0      ;;USER DEFINED
(3)      001166      000000      $TMP3:  .WORD      0      ;;USER DEFINED
(3)      001170      000000      $TMP4:  .WORD      0      ;;USER DEFINED
(3)      001172      000000      $TMP5:  .WORD      0      ;;USER DEFINED
(3)      001174      000000      $TMP6:  .WORD      0      ;;USER DEFINED
(3)      001176      000000      $TMP7:  .WORD      0      ;;USER DEFINED
(3)      001200      000000      $TMP10: .WORD      0      ;;USER DEFINED
(3)      001202      000000      $TMP11: .WORD      0      ;;USER DEFINED
(3)      001204      000000      $TMP12: .WORD      0      ;;USER DEFINED
(3)      001206      000000      $TMP13: .WORD      0      ;;USER DEFINED
(3)      001210      000000      $TMP14: .WORD      0      ;;USER DEFINED
(3)      001212      000000      $TMP15: .WORD      0      ;;USER DEFINED
(3)      001214      000000      $TMP16: .WORD      0      ;;USER DEFINED
(3)      001216      000000      $TMP17: .WORD      0      ;;USER DEFINED
(1)      001220      000000      $TIMES: 0      ;;MAX. NUMBER OF ITERATIONS
(1)      001222      000000      $ESCAPE: 0      ;;ESCAPE ON ERROR ADDRESS
(1)      001224      177607      000377      $BELL:  .ASCIZ <207><377><377>      ;;CODE FOR BELL
    
```

```

(1) 001230 077 $QUES: .ASCII /?/ ::QUESTION MARK
(1) 001231 015 $CRLF: .ASCII <15> ::CARRIAGE RETURN
(1) 001232 000012 $LF: .ASCII <12> ::LINE FEED
(2) ::*****
(2) $BTTL APT MAILBOX-ETABLE
(2)
(3) ::*****
(2) $EVEN
(2) 001234 $MAIL: ::APT MAILBOX
(2) 001234 000000 $MSGTY: .WORD AMSGTY ::MESSAGE TYPE CODE
(2) 001236 000000 $FATAL: .WORD AFATAL ::FATAL ERROR NUMBER
(2) 001240 000000 $TESTN: .WORD ATESTN ::TEST NUMBER
(2) 001242 000000 $PASS: .WORD APASS ::PASS COUNT
(2) 001244 000000 $DEVCT: .WORD ADEVCT ::DEVICE COUNT
(2) 001246 000000 $UNIT: .WORD AUNIT ::I/O UNIT NUMBER
(2) 001250 000000 $MSGAD: .WORD AMSGAD ::MESSAGE ADDRESS
(2) 001252 000000 $MSGLG: .WORD AMSGLG ::MESSAGE LENGTH
(2) 001254 $ETABLE: ::APT ENVIRONMENT TABLE
(2) 001254 000 $ENV: .BYTE AENV ::ENVIRONMENT BYTE
(2) 001255 000 $ENVM: .BYTE AENVM ::ENVIRONMENT MODE BITS
(2) 001256 000000 $SWREG: .WORD ASWREG ::APT SWITCH REGISTER
(2) 001260 000000 $USWR: .WORD AUSWR ::USER SWITCHES
(2) 001262 000000 $CPUOP: .WORD ACPUOP ::CPU TYPE,OPTIONS
(2) * BITS 15-11=CPU TYPE
(2) * 11/04=01,11/05=02,11/20=03,11/40-04,11/45=05
(2) * 11/70=06,PDQ=07,Q=10
(2) * BIT 10=REAL TIME CLOCK
(2) * BIT 9=FLOATING POINT PROCESSOR
(2) * BIT 8=MEMORY MANAGEMENT
(2) 001264 000 $MAMS1: .BYTE AMAMS1 ::HIGH ADDRESS,M.S. BYTE
(2) 001265 000 $MTYP1: .BYTE AMTYP1 ::MEM. TYPE,BLK#1
(2) * MEM. TYPE BYTE -- (HIGH BYTE)
(2) * 900 NSEC CORE=001
(2) * 300 NSEC BIPOLAR=002
(2) * 500 NSEC MOS=003
(2) 001266 000000 $MADR1: .WORD AMADR1 ::HIGH ADDRESS,BLK#1
(2) * MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
(2) 001270 000 $MAMS2: .BYTE AMAMS2 ::HIGH ADDRESS,M.S. BYTE
(2) 001271 000 $MTYP2: .BYTE AMTYP2 ::MEM. TYPE,BLK#2
(2) 001272 000000 $MADR2: .WORD AMADR2 ::MEM.LAST ADDRESS,BLK#2
(2) 001274 000 $MAMS3: .BYTE AMAMS3 ::HIGH ADDRESS,M.S.BYTE
(2) 001275 000 $MTYP3: .BYTE AMTYP3 ::MEM. TYPE,BLK#3
(2) 001276 000000 $MADR3: .WORD AMADR3 ::MEM.LAST ADDRESS,BLK#3
(2) 001300 000 $MAMS4: .BYTE AMAMS4 ::HIGH ADDRESS,M.S.BYTE
(2) 001301 000 $MTYP4: .BYTE AMTYP4 ::MEM. TYPE,BLK#4
(2) 001302 000000 $MADR4: .WORD AMADR4 ::MEM.LAST ADDRESS,BLK#4
(2) 001304 000000 $VECT1: .WORD AVECT1 ::INTERRUPT VECTOR#1,BUS PRIORITY#1
(2) 001306 000000 $VECT2: .WORD AVECT2 ::INTERRUPT VECTOR#2BUS PRIORITY#2
(2) 001310 000000 $BASE: .WORD ABASE ::BASE ADDRESS OF EQUIPMENT UNDER TEST
(2) 001312 $ETEND:
(2) $MEXIT
    
```

(1) .SBTTL ERROR POINTER TABLE
 (1)
 (1) ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 (1) ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 (1) ;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 (1) ;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 (1) ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
 (1)
 (1) ;* EM ;:POINTS TO THE ERROR MESSAGE
 (1) ;* DH ;:POINTS TO THE DATA HEADER
 (1) ;* DT ;:POINTS TO THE DATA
 (1) ;* DF ;:POINTS TO THE DATA FORMAT

(1) 001312 \$ERRTB:
 113 ;
 114 ; SET UP THE ERROR TABLE POINTERS.
 115 ; IF YOU ADD OR DELETE ANY ERRORS,
 116 ; DON'T FORGET TO REDEFINE THE TOTAL ERROR COUNT !..
 117 ;

118 000223 LASTEM= 223 ; 223 ERRORS TOTAL.

120	001312	044361	056340	057426	.WORD	EM1, DH1, DT1, DF1	; ERROR ITEM 1
(3)	001322	044411	056420	057450	.WORD	EM2, DH2, DT2, DF2	; ERROR ITEM 2
(3)	001332	044450	056506	057450	.WORD	EM3, DH3, DT3, DF3	; ERROR ITEM 3
(3)	001342	044502	056567	057450	.WORD	EM4, DH4, DT4, DF4	; ERROR ITEM 4
(3)	001352	044534	056653	057450	.WORD	EM5, DH5, DT5, DF5	; ERROR ITEM 5
(3)	001362	044570	056716	057472	.WORD	EM6, DH6, DT6, DF6	; ERROR ITEM 6
(3)	001372	044622	056716	057472	.WORD	EM7, DH7, DT7, DF7	; ERROR ITEM 7
(3)	001402	000000	000000	000000	.WORD	0,0,0,0	
(3)	001412	000000	000000	000000	.WORD	0,0,0,0	
(3)	001422	044655	056746	057450	.WORD	EM12, DH12, DT12, DF12	; ERROR ITEM 12
(3)	001432	000000	000000	000000	.WORD	0,0,0,0	
(3)	001442	000000	000000	000000	.WORD	0,0,0,0	
(3)	001452	000000	000000	000000	.WORD	0,0,0,0	
(3)	001462	044723	057027	057450	.WORD	EM16, DH16, DT16, DF16	; ERROR ITEM 16
(3)	001472	044763	056746	057450	.WORD	EM17, DH17, DT17, DF17	; ERROR ITEM 17
(3)	001502	045034	057077	057450	.WORD	EM20, DH20, DT20, DF20	; ERROR ITEM 20
(3)	001512	000000	000000	000000	.WORD	0,0,0,0	
(3)	001522	045105	056746	057450	.WORD	EM22, DH22, DT22, DF22	; ERROR ITEM 22
(3)	001532	045170	057077	057450	.WORD	EM23, DH23, DT23, DF23	; ERROR ITEM 23
(3)	001542	045253	057027	057450	.WORD	EM24, DH24, DT24, DF24	; ERROR ITEM 24
(3)	001552	045333	057156	057504	.WORD	EM25, DH25, DT25, DF25	; ERROR ITEM 25
(3)	001562	045403	057156	057532	.WORD	EM26, DH26, DT26, DF26	; ERROR ITEM 26
(3)	001572	000000	000000	000000	.WORD	0,0,0,0	
(3)	001602	000000	000000	000000	.WORD	0,0,0,0	
(3)	001612	045506	057156	057532	.WORD	EM31, DH31, DT31, DF31	; ERROR ITEM 31
(3)	001622	000000	000000	000000	.WORD	0,0,0,0	
(3)	001632	000000	000000	000000	.WORD	0,0,0,0	
(3)	001642	045611	057156	057554	.WORD	EM34, DH34, DT34, DF34	; ERROR ITEM 34
(3)	001652	045723	057156	057554	.WORD	EM35, DH35, DT35, DF35	; ERROR ITEM 35
(3)	001662	046035	057156	057554	.WORD	EM36, DH36, DT36, DF36	; ERROR ITEM 36
(3)	001672	046151	057156	057554	.WORD	EM37, DH37, DT37, DF37	; ERROR ITEM 37
(3)	001702	046265	057156	057554	.WORD	EM40, DH40, DT40, DF40	; ERROR ITEM 40
(3)	001712	046327	056746	057450	.WORD	EM41, DH41, DT41, DF41	; ERROR ITEM 41
(3)	001722	046376	057156	057532	.WORD	EM42, DH42, DT42, DF42	; ERROR ITEM 42

(3)	001732	046502	057156	057532	.WORD	EM43, DH43, DT43, DF43	: ERROR ITEM 43
(3)	001742	046606	057215	057602	.WORD	EM44, DH44, DT44, DF44	: ERROR ITEM 44
(3)	001752	000000	000000	000000	.WORD	0,0,0,0	
(3)	001762	046645	057215	057624	.WORD	EM46, DH46, DT46, DF46	: ERROR ITEM 46
(3)	001772	046717	057215	057472	.WORD	EM47, DH47, DT47, DF47	: ERROR ITEM 47
(3)	002002	047016	057215	057712	.WORD	EM50, DH50, DT50, DF50	: ERROR ITEM 50
(3)	002012	047124	057215	057712	.WORD	EM51, DH51, DT51, DF51	: ERROR ITEM 51
(3)	002022	047166	057156	057734	.WORD	EM52, DH52, DT52, DF52	: ERROR ITEM 52
(3)	002032	047222	057156	057532	.WORD	EM53, DH53, DT53, DF53	: ERROR ITEM 53
(3)	002042	047264	057156	057554	.WORD	EM54, DH54, DT54, DF54	: ERROR ITEM 54
(3)	002052	047330	057156	057734	.WORD	EM55, DH55, DT55, DF55	: ERROR ITEM 55
(3)	002062	047364	057156	057532	.WORD	EM56, DH56, DT56, DF56	: ERROR ITEM 56
(3)	002072	047426	057156	057554	.WORD	EM57, DH57, DT57, DF57	: ERROR ITEM 57
(3)	002102	047472	057156	057532	.WORD	EM60, DH60, DT60, DF60	: ERROR ITEM 60
(3)	002112	047534	057156	057554	.WORD	EM61, DH61, DT61, DF61	: ERROR ITEM 61
(3)	002122	047600	057156	057554	.WORD	EM62, DH62, DT62, DF62	: ERROR ITEM 62
(3)	002132	047716	057215	057712	.WORD	EM63, DH63, DT63, DF63	: ERROR ITEM 63
(3)	002142	050026	057246	057450	.WORD	EM64, DH64, DT64, DF64	: ERROR ITEM 64
(3)	002152	050070	057215	057472	.WORD	EM65, DH65, DT65, DF65	: ERROR ITEM 65
(3)	002162	050124	057215	057712	.WORD	EM66, DH66, DT66, DF66	: ERROR ITEM 66
(3)	002172	050170	057156	057734	.WORD	EM67, DH67, DT67, DF67	: ERROR ITEM 67
(3)	002202	050170	057156	057734	.WORD	EM70, DH70, DT70, DF70	: ERROR ITEM 70
(3)	002212	050273	057156	057554	.WORD	EM71, DH71, DT71, DF71	: ERROR ITEM 71
(3)	002222	050415	057156	057532	.WORD	EM72, DH72, DT72, DF72	: ERROR ITEM 72
(3)	002232	050460	057156	057554	.WORD	EM73, DH73, DT73, DF73	: ERROR ITEM 73
(3)	002242	050525	057156	057734	.WORD	EM74, DH74, DT74, DF74	: ERROR ITEM 74
(3)	002252	050525	057156	057734	.WORD	EM75, DH75, DT75, DF75	: ERROR ITEM 75
(3)	002262	050630	057156	057554	.WORD	EM76, DH76, DT76, DF76	: ERROR ITEM 76
(3)	002272	050752	057156	057532	.WORD	EM77, DH77, DT77, DF77	: ERROR ITEM 77
(3)	002302	051015	057156	057554	.WORD	EM100, DH100, DT100, DF100	: ERROR ITEM 100
(3)	002312	051062	057156	057734	.WORD	EM101, DH101, DT101, DF101	: ERROR ITEM 101
(3)	002322	051165	057156	057532	.WORD	EM102, DH102, DT102, DF102	: ERROR ITEM 102
(3)	002332	051231	057156	057554	.WORD	EM103, DH103, DT103, DF103	: ERROR ITEM 103
(3)	002342	051340	057156	057554	.WORD	EM104, DH104, DT104, DF104	: ERROR ITEM 104
(3)	002352	051406	057156	057734	.WORD	EM105, DH105, DT105, DF105	: ERROR ITEM 105
(3)	002362	051512	057156	057532	.WORD	EM106, DH106, DT106, DF106	: ERROR ITEM 106
(3)	002372	051557	057156	057554	.WORD	EM107, DH107, DT107, DF107	: ERROR ITEM 107
(3)	002402	051674	057156	057554	.WORD	EM110, DH110, DT110, DF110	: ERROR ITEM 110
(3)	002412	051743	057215	057472	.WORD	EM111, DH111, DT111, DF111	: ERROR ITEM 111
(3)	002422	052016	057215	057472	.WORD	EM112, DH112, DT112, DF112	: ERROR ITEM 112
(3)	002432	052072	056746	057450	.WORD	EM113, DH113, DT113, DF113	: ERROR ITEM 113
(3)	002442	052137	057077	057450	.WORD	EM114, DH114, DT114, DF114	: ERROR ITEM 114
(3)	002452	052204	056716	057472	.WORD	EM115, DH115, DT115, DF115	: ERROR ITEM 115
(3)	002462	000000	000000	000000	.WORD	0,0,0,0	
(3)	002472	000000	000000	000000	.WORD	0,0,0,0	
(3)	002502	000000	000000	000000	.WORD	0,0,0,0	
(3)	002512	052266	057215	057472	.WORD	EM121, DH121, DT121, DF121	: ERROR ITEM 121
(3)	002522	052365	057215	057712	.WORD	EM122, DH122, DT122, DF122	: ERROR ITEM 122
(3)	002532	052473	057215	057712	.WORD	EM123, DH123, DT123, DF123	: ERROR ITEM 123
(3)	002542	000000	000000	000000	.WORD	0,0,0,0	
(3)	002552	052535	056746	057752	.WORD	EM125, DH125, DT125, DF125	: ERROR ITEM 125
(3)	002562	052641	057215	057472	.WORD	EM126, DH126, DT126, DF126	: ERROR ITEM 126
(3)	002572	052762	057215	057472	.WORD	EM127, DH127, DT127, DF127	: ERROR ITEM 127
(3)	002602	053100	057077	057752	.WORD	EM130, DH130, DT130, DF130	: ERROR ITEM 130
(3)	002612	000000	000000	000000	.WORD	0,0,0,0	
(3)	002622	000000	000000	000000	.WORD	0,0,0,0	

(3)	002632	000000	000000	000000	.WORD	0,0,0,0			
(3)	002642	000000	000000	000000	.WORD	0,0,0,0			
(3)	002652	000000	000000	000000	.WORD	0,0,0,0			
(3)	002662	000000	000000	000000	.WORD	0,0,0,0			
(3)	002672	000000	000000	000000	.WORD	0,0,0,0			
(3)	002702	053144	056746	060000	.WORD	EM140, DH140, DT140, DF140	:	ERROR ITEM 140	
(3)	002712	053225	056746	060000	.WORD	EM141, DH141, DT141, DF141	:	ERROR ITEM 141	
(3)	002722	000000	000000	000000	.WORD	0,0,0,0			
(3)	002732	000000	000000	000000	.WORD	0,0,0,0			
(3)	002742	000000	000000	000000	.WORD	0,0,0,0			
(3)	002752	053306	057324	060042	.WORD	EM145, DH145, DT145, DF145	:	ERROR ITEM 145	
(3)	002762	053345	057324	060042	.WORD	EM146, DH146, DT146, DF146	:	ERROR ITEM 146	
(3)	002772	053420	057324	060042	.WORD	EM147, DH147, DT147, DF147	:	ERROR ITEM 147	
(3)	003002	053470	057324	060042	.WORD	EM150, DH150, DT150, DF150	:	ERROR ITEM 150	
(3)	003012	053531	057324	060042	.WORD	EM151, DH151, DT151, DF151	:	ERROR ITEM 151	
(3)	003022	053572	057324	060042	.WORD	EM152, DH152, DT152, DF152	:	ERROR ITEM 152	
(3)	003032	053714	057324	060042	.WORD	EM153, DH153, DT153, DF153	:	ERROR ITEM 153	
(3)	003042	054036	057324	060042	.WORD	EM154, DH154, DT154, DF154	:	ERROR ITEM 154	
(3)	003052	054076	057324	060042	.WORD	EM155, DH155, DT155, DF155	:	ERROR ITEM 155	
(3)	003062	054145	057324	060042	.WORD	EM156, DH156, DT156, DF156	:	ERROR ITEM 156	
(3)	003072	000000	000000	000000	.WORD	0,0,0,0			
(3)	003102	054255	056746	060000	.WORD	EM160, DH160, DT160, DF160	:	ERROR ITEM 160	
(3)	003112	000000	000000	000000	.WORD	0,0,0,0			
(3)	003122	054376	056746	060000	.WORD	EM162, DH162, DT162, DF162	:	ERROR ITEM 162	
(3)	003132	054447	056746	060000	.WORD	EM163, DH163, DT163, DF163	:	ERROR ITEM 163	
(3)	003142	054522	056746	060000	.WORD	EM164, DH164, DT164, DF164	:	ERROR ITEM 164	
(3)	003152	054602	056716	060100	.WORD	EM165, DH165, DT165, DF165	:	ERROR ITEM 165	
(3)	003162	054656	056746	060000	.WORD	EM166, DH166, DT166, DF166	:	ERROR ITEM 166	
(3)	003172	054730	056746	060000	.WORD	EM167, DH167, DT167, DF167	:	ERROR ITEM 167	
(3)	003202	055004	056746	060000	.WORD	EM170, DH170, DT170, DF170	:	ERROR ITEM 170	
(3)	003212	055065	056716	060100	.WORD	EM171, DH171, DT171, DF171	:	ERROR ITEM 171	
(3)	003222	055142	057156	057532	.WORD	EM172, DH172, DT172, DF172	:	ERROR ITEM 172	
(3)	003232	055206	057324	060126	.WORD	EM173, DH173, DT173, DF173	:	ERROR ITEM 173	
(3)	003242	000000	000000	000000	.WORD	0,0,0,0			
(3)	003252	000000	000000	000000	.WORD	0,0,0,0			
(3)	003262	000000	000000	000000	.WORD	0,0,0,0			
(3)	003272	000000	000000	000000	.WORD	0,0,0,0			
(3)	003302	000000	000000	000000	.WORD	0,0,0,0			
(3)	003312	055254	056746	060160	.WORD	EM201, DH201, DT201, DF201	:	ERROR ITEM 201	
(3)	003322	055343	056746	060212	.WORD	EM202, DH202, DT202, DF202	:	ERROR ITEM 202	
(3)	003332	055405	057215	060244	.WORD	EM203, DH203, DT203, DF203	:	ERROR ITEM 203	
(3)	003342	055463	057362	060266	.WORD	EM204, DH204, DT204, DF204	:	ERROR ITEM 204	
(3)	003352	055537	057362	060266	.WORD	EM205, DH205, DT205, DF205	:	ERROR ITEM 205	
(3)	003362	055607	056746	060310	.WORD	EM206, DH206, DT206, DF206	:	ERROR ITEM 206	
(3)	003372	055664	056746	060310	.WORD	EM207, DH207, DT207, DF207	:	ERROR ITEM 207	
(3)	003402	055773	056746	060310	.WORD	EM210, DH210, DT210, DF210	:	ERROR ITEM 210	
(3)	003412	056050	056746	060310	.WORD	EM211, DH211, DT211, DF211	:	ERROR ITEM 211	
(3)	003422	000000	000000	000000	.WORD	0,0,0,0			
(3)	003432	000000	000000	000000	.WORD	0,0,0,0			
(3)	003442	000000	000000	000000	.WORD	0,0,0,0			
(3)	003452	000000	000000	000000	.WORD	0,0,0,0			
(3)	003462	000000	000000	000000	.WORD	0,0,0,0			
(3)	003472	000000	000000	000000	.WORD	0,0,0,0			
(3)	003502	056157	056746	060352	.WORD	EM220, DH220, DT220, DF220	:	ERROR ITEM 220	
(3)	003512	056232	056746	060352	.WORD	EM221, DH221, DT221, DF221	:	ERROR ITEM 221	
(3)	003522	000000	000000	000000	.WORD	0,0,0,0			

(3) 003532 000000 000000 000000 .WORD 0,0,0,0

```

122
123      ; INITIAL START/RESTART HERE.
124
126 003542      START:RESTART:
127 003542 000240      NOP
128      .SBTTL INITIALIZE THE COMMON TAGS
(1)      ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
(1) 003544 012706 001100      MOV      #$CMTAG,R6      ;:FIRST LOCATION TO BE CLEARED
(1) 003550 005026      CLR      (R6)+      ;:CLEAR MEMORY LOCATION
(1) 003552 022706 001140      CMP      #SWR,R6 ;:DONE?
(1) 003556 001374      BNE      -6      ;:LOOP BACK IF NO
(1) 003560 012706 001100      MOV      #STACK,SP      ;:SETUP THE STACK POINTER
(1)      ;:INITIALIZE A FEW VECTORS
(1) 003564 012737 037504 000020      MOV      #$SCOPE,@#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
(1) 003572 012737 000340 000022      MOV      #340,@#IOTVEC+2 ;:LEVEL 7
(1) 003600 012737 042542 000030      MOV      #ERROR,@#EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
(1) 003606 012737 000340 000032      MOV      #340,@#EMTVEC+2 ;:LEVEL 7
(1) 003614 012737 042220 000034      MOV      #STRAP,@#TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
(1) 003622 012737 000340 000036      MOV      #340,@#TRAPVEC+2 ;:LEVEL 7
(1) 003630 012737 042370 000024      MOV      #SPWRDN,@#PURVEC ;:POWER FAILURE VECTOR
(1) 003636 012737 000340 000026      MOV      #340,@#PURVEC+2 ;:LEVEL 7
(1) 003644 013737 037330 037322      MOV      $ENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
(1) 003652 005037 001220      CLR      $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
(1) 003656 005037 001222      CLR      $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
(1) 003662 112737 000001 001115      MOV      #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
(1) 003670 012737 003670 001106      MOV      #,$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1) 003676 012737 003676 001110      MOV      #,$SLPERR ;:SETUP THE ERROR LOOP ADDRESS
(2)      ;:SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)      ;:EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
(2) 003704 013746 000004      MOV      @#ERRVEC,-(SP) ;:SAVE ERROR VECTOR
(2) 003710 012737 003744 000004      MOV      #64$,@#ERRVEC ;:SET UP ERROR VECTOR
(2) 003716 012737 177570 001140      MOV      #DSWR,SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
(2) 003724 012737 177570 001142      MOV      #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
(2) 003732 022777 177777 175200      CMP      #-1,@SWR ;:TRY TO REFERENCE HARDWARE SWR
(2) 003740 001012      BNE      66$ ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)      ;:AND THE HARDWARE SWR IS NOT - -1
(2) 003742 000403      BR      65$ ;:BRANCH IF NO TIMEOUT
(2) 003744 012716 003752      64$: MOV      #65$,(SP) ;:SET UP FOR TRAP RETURN
(2) 003750 000002      RTI
(2) 003752 012737 000176 001140      65$: MOV      #SWREG,SWR ;:POINT TO SOFTWARE SWR
(2) 003760 012737 000174 001142      MOV      #DISPREG,DISPLAY
(2) 003766 012637 000004      66$: MOV      (SP)+,@#ERRVEC ;:RESTORE ERROR VECTOR
(1)
(2) 003772 005037 001242      CLR      $PASS ;:CLEAR PASS COUNT
(2) 003776 132737 000200 001255      BITB   #APTSIZE,$ENVM ;:TEST USER SIZE UNDER APT
(2) 004004 001403      BEQ      67$ ;:YES,USE NON-APT SWITCH
(2) 004006 012737 001256 001140      MOV      #SWREG,SWR ;:NO,USE APT SWITCH REGISTER
(2) 004014      67$:

```

```

130
131
132
133
134 004014 005227 177777
135 004020 001032
136 004022 023727 000042 037464
137 004030 001426
138 004032 104401 004040
(1) 004036 000423
(1)
(1) 004106
139 004106
(1)
(1) 004106 005737 000042
(1) 004112 001012
(1) 004114 123727 001254 000001
(1) 004122 001406
(1) 004124 023727 001140 000176
(1) 004132 001005
(1) 004134 104406
(1) 004136 000403
(1) 004140 112737 000001 001134
(1) 004146
140
141
142
143 004146 012706 001100
144 004152 012737 004200 000244
145 004160 012737 004234 000010
146 004166 012737 004224 000004
147 004174 000137 004356

:
: TYPE NAME ON INITIAL PASS IF NOT ACT MODE.
: GET SWR IF NOT AUTO MODE (ACT, APT, OR XXDP CHAIN).
:
: INC # -1 ; INITIAL PASS ??
: BNE 1$ ; NO
: CMP 42,$SENDAD ; ACT-11 ??
: BEQ 1$ ; YES
: TYPE ,69$ ' ; TYPE ASCIZ STRING
: BR 68$ ; GET OVER THE ASCIZ
:
: 69$: .ASCIZ <CRLF>'CJFPAA -- FPF11 DIAGNOSTIC, PART 1'<CRLF>
: 68$:
: 1$:
:
: .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
: TST @#42 ; ARE WE RUNNING UNDER XXDP/ACT?
: BNE 70$ ; BRANCH IF YES
: CMPB $ENV,#1 ; ARE WE RUNNING UNDER APT?
: BEQ 70$ ; BRANCH IF YES
: CMP SWR,#SWREG ; SOFTWARE SWITCH REG SELECTED?
: BNE 71$ ; BRANCH IF NO
: GTSWR ; GET SOFT-SWR SETTINGS
: BR 71$
: 70$: MOVB #1,$AUTOB ; SET AUTO-MODE INDICATOR
: 71$:
:
: CONTINUE HERE AFTER 'END-PASS'.
:
: LOOP: MOV #STACK,SP ; RESET STACK POINTER.
: MOV #TRP244,FPVEC ; RESET INTERRUPT VECTORS.
: MOV #TRP10,RESVEC
: MOV #TRP04,ERRVEC
: JMP TST1 ; GO START 'EM UP !!!!
    
```

```

149      .SBTTL  COMMON SUBROUTINES
150      :
151      : THESE HANDLE UNEXPECTED TRAPS TO 244, 10, AND 4.
152      : REPORT APPROPRIATE ERROR AND ABORT THE CURRENT TEST.
153      :
154      .ENABL  LSB
155 004200 011637 001164 TRP244: MOV      (SP),@#$TMP2      ; GET PC OF TRAP.
156 004204 170200      STFPS   R0              ; GET FPS
157 004206 010037 001166      MOV      R0,@#$TMP3
158 004212 170300      STST    R0              ; GET FEC
159 004214 010037 001170      MOV      R0,@#$TMP4
160 004220 104005      ERROR   5
161 004222 000407      BR       1$              ; COMMON EXIT.
162
163 004224 011637 001164 TRP04: MOV      (SP),@#$TMP2      ; GET PC OF TRAP.
164 004230 104006      ERROR   6
165 004232 000403      BR       1$
166
167 004234 011637 001164 TRP10: MOV      (SP),@#$TMP2      ; GET PC OF TRAP.
168 004240 104007      ERROR   7
169
170 004242 022626      1$:    CMP      (SP)+,(SP)+      ; COMMON EXIT, FIX STACK...
171 004244 104401 043341      TYPE     ,ABORT          ; ...AND TELL THE MAN.
172 004250 123727 001102 000051  CMPB    $STNM,#LASTST    ; ON THE LAST TEST ??
173 004256 001002      BNE     2$              ; NO.
174 004260 000137 037274      JMP     $EOP            ; YES, END-PASS.
175 004264 113700 001102      2$:    MOVB   $STNM,R0      ; GET TEST NUM...
176 004270 006300      ASL    R0              ; ...SHIFT TO WORD INDEX.
177 004272 016001 040022      MOV    $$W08TBL(R0),R1 ; GET NEXT TEST ADDRESS...
178 004276 000161 177776      JMP    -2(R1)          ; ...AND GO THFRE.
*79      .DSABL  LSB
    
```

```

181
182
183
184
185
186
187
188
189 004302 012737 000002 004352 CHECK2: .ENABL  LSB
190 004310 000403          SKP3          ; 32 BIT COMPARE.
191 004312 012737 000004 004352 CHECK4: MOV      #4,3$      ; 64 BIT COMPARE.
192 004320 010046          MOV      R0,-(SP)   ; SAVE REGISTERS.
193 004322 010146          MOV      R1,-(SP)
194 004324 012570          MOV      (R5)+,R0   ; SET OPERAND ADDRESSES.
195 004326 012501          MOV      (R5)+,R1
196 004330 000240          NOP
197 004332 022021          1$:  CMP      (R0)+,(R1)+ ; COMPARE.
198 004334 001003          BNE      2$         ; EXIT IF NOT EQUAL.
199 004336 005337 004352          DEC      3$
200 004342 001373          BNE      1$
201 004344 012601          2$:  MOV      (SP)+,R1   ; RESTORE REGISTERS.
202 004346 012600          MOV      (SP)+,R0
203 004350 005727          TST      (PC)+
204 004352 000000          3$:  0             ; 0 IF EQUAL, NZ OTHERWISE.
205 004354 000205          RTS      R5
206          .DSABL  LSB
    
```

```
208 .SBTTL
209 .SBTTL          FPF PART 1 TESTS
210 .SBTTL
232 :*****
(3) *TEST 1          LDFPS, STFPS, AND DATA PATHS -- SRC AND DST MODE 0
(4) *
(4) *THIS IS A TEST OF THE LDFPS (LOAD FP STATUS) AND STFPS
(4) *(STORE FP STATUS) INSTRUCTIONS. A COUNT PATTERN IS GENERATED
(4) *AND RUN THROUGH THE FLOATING POINT STATUS REGISTER.
(4) *THIS WILL TEST THE 16-BIT TRI STATE BUS WHICH CONNECTS THE CPU
(4) *WITH THE FPP AND ALSO RUNS INTERNALLY WITHIN THE FPP.
(4) *SRC/DST MODE 0 IS USED.
(4) *NOTE THAT BITS 12 AND 13 ARE UNUSED IN F;S AND ARE MASKED OFF.
(4) *
(4) *TO PREVENT LOCKING OUT THE COMPLETION OF THE TEST BECAUSE OF
(4) *AN EXCESSIVE NUMBER OF ERRORS, ONLY THE FIRST FIVE ERRORS
(4) *WILL BE REPORTED THEN THE TEST WILL BE COMPLETED AND
(4) *AN ERROR SUMMARY GIVEN, AS FOLLOWS:
(4) *
(4) *A DYNAMIC RECORD OF THE LOGICAL 'AND' AND 'OR' OF THE FAILING
(4) *DATA PATTERNS IS KEPT. THESE CAN BE VERY USEFUL IN DETERMINING
(4) *STUCK BITS. WHEN THE TEST COMPLETES, THIS SUMMARY IS PRINTED
(4) *IF ANY ERRORS WERE ENCOUNTERED.
(4) *
(3) :*****
(2) 004356 000004 TST1: SCOPE
233 004360 005000 1$: CLR R0 ; INITIALIZE COUNT PATTERN.
234 004362 012702 177777 MOV #-1,R2 ; R2 IS THE 'AND' OF BAD DATA.
235 004366 005003 CLR R3 ; R3 IS THE 'OR' OF BAD DATA.
236 004370 012737 004406 001164 MOV #A2,$TMP2 ; ERROR PC.
237 004376 005037 001200 CLR $TMP10 ; ERROR COUNTER.
238 004402 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.
239 004404 010004 A1: MOV R0,R4
240 004406 170104 A2: LDFPS R4 ; R4 => FPS...
241 004410 012701 177777 MOV #-1,R1
242 004414 170201 STFPS R1 ;...AND FPS => R1.
243 004416 042704 030000 BIC #30000,R4 ; MASK OFF EXPD DATA <13:12>.
244 004422 020401 CMP R4,R1 ; CHECK EXP'D VS REC'D
245 004424 001421 BEQ 4$ ; BR IF OK.
246
247 004426 050103 3$: BIS R1,R3 ;MAKE 'OR' OF FAILING DATA.
248 004430 005101 COM R1
249 004432 040102 BIC R1,R2 ;MAKE 'AND' OF FAILING DATA.
250 004434 005101 COM R1
251 004436 023727 001200 000005 CMP $TMP10,#5 ; 5 ERRORS LOGGED ??
252 004444 103011 BHIS 4$ ; YES, DON'T REPORT ANY MORE.
253 004446 005237 001200 INC $TMP10 ; NOT YET, REPORT ERROR.
254 004452 010037 001166 MOV R0,@$TMP3 ; WROTE
255 004456 010137 001170 MOV R1,@$TMP4 ; READ
256 004462 010437 001172 MOV R4,@$TMP5 ; EXPD
257 004466 104001 ERROR 1
258 004470 077033 4$: SOB R0,A1 ; LOOP UNTIL DONE.
259
260 004472 005737 001200 5$: TST $TMP10 ; ALL DONE, ANY ERRORS ??
261 004476 001410 BEQ A3 ; BR IF NOT.
262 004500 013737 001200 001164 MOV $TMP10,$TMP2 ; OTHERWISE, GIVE AND/OR SUMMARY.
```

263 004506 010237 001166
264 004512 010337 001170
265 004516 104002
266
267
268
269
270
271
272 004520 170106
273 004522 000240 000240
274 004526 170107
275 004530 000240 000240
276 004534
(1) 004534 104412
(3) 004536 000400

MOV R2,\$TMP3 ; 'AND' SUMMARY.
MOV R3,\$TMP4 ; 'OR' SUMMARY.
ERROR 2
:
: NOW THE PC (R7) AND SP (R6) ARE VALID SRC ADDRESSES.
: CAN'T THINK WHY ANYONE WOULD WANT TO PUT THE PC OR STACK POINTER
: IN THE FPS -- BUT DO IT ANYWAY JUST TO TICK THAT MICROSTATE, AND
: SEE THAT IT DOESN'T TRAP FOR ANY REASON.
:
A3: LDFPS SP ; R6 => FPS
240,240 ; WE'RE HERE IF IT DIDN'T TRAP.
A4: LDFPS PC ; R7 => FPS
240,240 ; DITTO
ADONE:
CLRFPS ;; CLEAR FP STATUS...
BR TST2 ;...AND PROCEED.


```
286 (3) :*****  
(4) :*TEST 2 COPY CONDITION CODES  
(4) :*  
(4) :*THIS IS A TEST OF THE COPY CONDITION CODES INSTRUCTION, CFCC.  
(4) :*INSURE THAT ONLY PSW<3:0> ARE AFFECTED BY THE CFCC, AND THAT  
(4) :*THE FP STATUS REGISTER IS UNALTERED.  
(4) :* NOTE: THE CPU IS IN 'USER MODE' DURING THIS TEST.  
(3) :*  
(2) :*****  
287 004540 000004 TST2: SCOPE ; FPS CODE PATTERN.  
004542 012700 000017 MOV #17,R0 ; SET PC IN CASE OF ERROR.  
288 004546 012737 0C4600 001164 MOV #B2,$TMP2 ; LOOP HERE ON ERROR IF SWR9 1.  
289 004554 104411 LUPERR ; SET 'USER MODE, PRI 7'.  
290 004556 052737 140340 177776 B1: BIS #140340,PSW ; SET FPS.  
291 004564 170100 LDFPS R0 ;  
292 004566 013701 177776 MOV PSW,R1 ; SET EXP'D PSW => R1.  
293 004572 042701 000017 BIC #17,R1 ; XCT 'COPY CONDITION CODES'  
294 004576 050001 BIS R0,R1 ;  
295 004600 170000 B2: CFCC ; GET FINAL PSW...  
296 004602 000240 240 ; ...AND FPS.  
297 004604 013702 177776 MOV PSW,R2 ;  
298 004610 170203 STFPS R3 ;  
299  
300 004612 020102 CMP R1,R2 ; PSW RIGHT ??  
301 004614 001405 BEQ 1$ ; BR IF SO.  
302 004616 010137 001170 MOV R1,$TMP4 ;  
303 004622 010237 001166 MOV R2,$TMP3 ;  
304 004626 104003 ERROR 3 ; *** PSW WRONG ***  
305  
306 004630 020003 1$: CMP R0,R3 ; FPS RIGHT ??  
307 004632 001405 BEQ 2$ ; BR IF SO.  
308 004634 010037 001166 MOV R0,$TMP3 ;  
309 004640 010337 001170 MOV R3,$TMP4 ;  
310 004644 104004 ERROR 4 ; *** FPS ALTERED ***  
311  
312 004646 005300 2$: DEC R0 ; NEXT CODE PATTERN.  
313 004650 100342 BPL B1 ; LOOP 'TIL DONE.  
314  
315 004652 005037 177776 BDONE: CLR PSW ; RESTORE KERNAL MODE, PRI 0.  
316 004656 104412 CLRFPS ; CLEAR FP STATUS...  
(3) 004660 000400 BR TST3 ; ...AND PROCEED.
```

326
 (3)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2)
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 (1)
 368
 369
 370
 371

004662 000004
 004664 112737 000106 044704
 004672 012737 170001 005036
 004700 005002
 004702 012703 147577
 004706 004737 005022
 004712 112737 000104 044704
 004720 012737 170011 005036
 004726 012702 000200
 004732 012703 147777
 004736 004737 005022
 004742 112737 000111 044704
 004750 012737 170002 005036
 004756 005002
 004760 012703 147677
 004764 004737 005022
 004770 112737 000114 044704
 004776 012737 170012 005036
 005004 012702 000100
 005010 012703 147777
 005014 004737 005022
 005020 000443
 005022 013737 005036 005076
 005030 104411
 005032 005000
 005034 170100
 005036 170001
 005040 170201
 005042 020102
 005044 001410
 005046 012737 005036 001164
 005054 010137 001166
 005060 010237 001170
 005064 104012
 005066
 005066 104411
 005070 012700 147777
 005074 170100
 005076 170001
 005100 170201

```

*****
*TEST 3      SETF, SETD, SETI, AND SETL
*
*THIS IS A TEST OF THE SETF, SETD, SETI AND SETL INSTRUCTIONS.
*EACH INSTRUCTION IS EXECUTED WITH THE FPS CLEAR, AND THEN
*AGAIN WITH THE FPS CONTAINING ALL 1'S (147777).
*THE RESULTING FPS IS CHECKED IN EACH CASE.
*
*****
TST3:  SCOPE
      MOV  #F,EM12X
      MOV  #SETF,C15
      CLR  R2
      MOV  #147577,R3
      JSR  PC,C1
      ; EXPECT BOTH CLEAR 1ST...
      ; ...THEN FD ONLY.
      ; TEST 'SETF'.

      MOV  #D,EM12X
      MOV  #SETD,C15
      MOV  #200,R2
      MOV  #147777,R3
      JSR  PC,C1
      ; EXPECT FD - 1...
      ; ...THEN ALL.
      ; TEST 'SETD'.

      MOV  #I,EM12X
      MOV  #SETI,C15
      CLR  R2
      MOV  #147677,R3
      JSR  PC,C1
      ; EXPECT BOTH CLEAR...
      ; ...THEN FL ONLY.
      ; TEST 'SETI'.

      MOV  #L,EM12X
      MOV  #SETL,C15
      MOV  #100,R2
      MOV  #147777,R3
      JSR  PC,C1
      BR   CDONE
      ; EXPECT FL ONLY...
      ; ...THEN ALL.
      ; TEST 'SETL'.

; CLEAR FPS AND TEST SET_, THEN SET 1'S AND TEST AGAIN.
C1:   MOV  C15,C25
      LUPERR
      CLR  R0
      LDFPS R0
      ; LOOP HERE ON ERROR IF SWR9 = 1.
      ; 0'S TO FPS.
      ; TEST INSTRUCTION.
      ; GET RESULT.
C15:  STFPS R1
      CMP  R1,R2
      BEQ  1$
      ; BR IF 1ST PASS OK.
      MOV  #C15,$TMP2
      MOV  R1,$TMP3
      MOV  R2,$TMP4
      ERROR 12
      ; *** SET_ FAILED ***

1$:   LUPERR
      MOV  #147777,R0
      ; LOOP HERE ON ERROR IF SWR9 = 1.
      ; 1'S TO FPS.
      ; TEST INSTRUCTION AGAIN.
      ; GET RESULT
C25:  SETF
      STFPS R1
  
```

```
372 005102 020103      CMP      R1,R3
373 005104 001410      BEQ      1$          ; BR IF 2ND PASS OK.
374 005106 012737 005076 001164      MOV      #C25,$TMP2
375 005114 010137 001166      MOV      R1,$TMP3
376 005120 010337 001170      MOV      R3,$TMP4
377 005124 104012      ERROR    12          ; *** SET_ FAILED ***
378 005126 000207      RTS      PC
379
380 005130      CDONE:      CLRFP5
(1) 005130 104412      BR      TST4
(3) 005132 000400      ;: CLEAR FP STATUS...
;: ...AND PROCEED.
```

```

392 (3) *****
(4) *TEST 4      ILLEGAL FPP OP-CODE TRAP, AND STST -- DST MODE 0
(4) *
(4) *THIS IS A TEST OF THE ILLEGAL FP OP-CODE TRAP FUNCTION.
(4) *FP OPCODES 170003 THRU 170010 AND 170013 THRU 170077 ARE
(4) *ILLEGAL (INVALID) AND SHOULD CAUSE A TRAP TO 244 (WHEN
(4) *INTERRUPTS ARE ENABLED).
(4) *WHEN ANY OF THE ABOVE OPCODES TRAP, THE FEC REGISTER SHOULD
(4) *BE SET TO CODE 2, TO INDICATE AN OPCODE ERROR.
(4) *
(3) *****
(2) 005134 000004          TST4:  SCOPF
393 005136 012705 170003      MOV      #170003,R5      ; INITIAL OP CODE.
394 005142 104411          LUPERR                    ; : LOOP HERE ON ERROR IF SWR9  1.
395 005144 012737 005242 000244 D1:  MOV      #D244,@#FVVEC  ;
396 005152 005002          CLR      R2              ; FOR OBSERVED FEC...
397 005154 005001          CLR      R1              ; ...AND FPS.
398 005156 005000          CLR      R0
399 005160 170100          LDFPS   R0              ; CLEAR FPS (INTERRUPT ENA).
400 005162 010537 005174      MOV      R5,@#D2        ; SET AN ILLEGAL INSTRUCTION.
401 005166 012737 005174 001164 D2:  MOV      #D2,@#STMP2
402 005174 170000          170000                    ; ILLEGAL OPCODES EXECUTED HERE.
403 005176 000240          240
404
405 005200 170201          STFPS   R1              ; IT DIDN'T TRAP, GET FPS.
406 005202 013737 005174 001166 MOV      D2,STMP3
407 005210 010137 001170      MOV      R1,@#STMP4
408 005214 104016          ERROR   16              ; *** OPCODE DIDN'T TRAP ***
409
410 005216 020527 170077      D3:  CMP      R5,#170077    ; LAST OPCODE DONE ??
411 005222 001435          BEQ     DDONE            ; BR IF SO.
412 005224 005205          INC     R5              ; NO, SET NEXT ONE.
413 005226 020527 170011      CMP      R5,#170011
414 005232 001002          BNE     1$              ; LOOP FOR 03 THRU 10...
415 005234 012705 170013      MOV      #170013,R5
416 005240 000741          1$:  BR      D1              ; ...AND 13 THRU 77.
417
418 ; WE'RE HERE ON TRAP TO 244.
419
420 005242 022626          D244:  CMP      (SP)+,(SP)+  ; FIX STACK.
421 005244 170201          STFPS   R1              ; GET THE FPS AND CHECK IT.
422 005246 020127 100000      CMP      R1,#BIT15
423 005252 001406          BEQ     D4              ; BR IF FPS IS OK.
424 005254 010137 001166      MOV      R1,@#STMP3
425 005260 012737 100000 001170 MOV      #BIT15,STMP4
426 005266 104017          ERROR   17              ; *** FPS WRONG ***
427
428 005270 170302          D4:  STST   R2              ; GET THE FEC.
429 005272 020227 000002      CMP      R2,#2
430 005276 001406          BEQ     1$              ; BR IF FEC IS RIGHT.
431 005300 010237 001166      MOV      R2,@#STMP3
432 005304 012737 000002 001170 MOV      #2,@#STMP4
433 005312 104020          ERROR   20              ; *** FEC WRONG ***
434 005314 000740          1$:  BR      D3
435
436 005316          DDONE:

```

CJFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJFPAA.P11 28-JAN-81 09:50 T4

MACY11 30G(1063) ^{K 3} 28-JAN-81 10:06 PAGE 2-6
ILLEGAL FPP OP-CODE TRAP, AND STST -- DST MODE 0

SEQ 0036

(1) 005316 104412
(3) 005320 000400

CLRFPS :: CLEAR FP STATUS...
BR TST5 ::...AND PROCEED.

```

445
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 005322 000004
446 005324 012737 005432 000244
447 005332 104411
448 005334 005002
449 005336 005001
450 005340 012700 040000
451 005344 170100
452 005346 012737 005354 001164
453 005354 170077
454 005356 000240
455
456 005360 170201
457 005362 020127 140000
458 005366 001406
459 005370 010137 001166
460 005374 012737 140000 001170
461 005402 104022
462
463 005404 170302
464 005406 020227 000002
465 005412 001406
466 005414 010237 001166
467 005420 012737 000002 001170
468 005426 104023
469 005430 000412
470
471
472
473 005432 011637 001164
474 005436 022626
475 005440 013737 005354 001166
476 005446 170201
477 005450 010137 001170
478 005454 104024
479
480 005456
(1) 005456 104412
(3) 005460 000400

*****
*TEST 5 INTERRUPT DISABLE (FID) BIT
*
*THIS IS A TEST OF THE INTERRUPT DISABLE (FID) BIT.
*AN ILLEGAL INSTRUCTION IS EXECUTED WITH FID - 1.
*NO INTERRUPT SHOULD OCCUR.
*
*****
TST5: SCOPE
MOV #E244,@#FPVEC ;SETUP FOR THE INTERRUPT.
LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
E1: CLR R2 ; FOR OBSERVED FEC...
CLR R1 ;...AND FPS.
MOV #BIT14,R0
LDFPS R0 ; SET FID BIT.
E2: MOV #E2,@#STMP2
170077 ; XCT ILLEGAL OPCODE.
240
STFPS R1 ; GET FPS.
CMP R1,#140000
BEQ 1$ ; BR IF FPS IS RIGHT.
MOV R1,$STMP3
MOV #140000,$STMP4
ERROR 22 ; *** FPS WRONG ***
1$: STST R2 ; GET FEC.
CMP R2,#2
BEQ 2$ ; BR IF FEC IS RIGHT.
MOV R2,$STMP3
MOV #2,$STMP4
ERROR 23 ; *** FEC WRONG ***
2$: BR EDONE
: WE'RE HERE ON A TRAP -- FID = 1, WHAT THE HELL ?????
E244: MOV (SP),$STMP2 ; SET TRAP PC.
CMP (SP)+,(SP)+
MOV E2,$STMP3
STFPS R1
MOV R1,$STMP4
ERROR 24 ; *** TRAPPED WITH FID = 1 ***
EDONE: CLRFPS ;: CLEAR FP STATUS...
BR TST6 ;:...AND PROCEED.

```

```
492 :*****
(3) :*TEST 6 LDD AND STD -- FSRC AND FDST MODE 1
(4) :*
(4) :*THIS IS A TEST OF THE LOAD AND STORE DOUBLE INSTRUCTIONS.
(4) :* LDD (R0),ACO AND STD ACO,(R0)
(4) :*
(4) :*NOTE THAT THIS IS THE FIRST ATTEMPT TO USE A'0. THIS IMPLIES
(4) :*THAT IT MAY NOT BE POSSIBLE TO ISOLATE A FAILURE TO EITHER THE
(4) :*AC ITSELF OR THE MICROCODE FLOW -- BUT WE'LL GIVE IT A TRY.
(4) :*
(3) :*****
(2) 005462 000004 TST6: SCOPE
493 :
494 : TEST THE LOAD FIRST.
495 :
496 005464 F1: LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
(1) 005464 104411 SETD ;: SET DOUBLE.
497 005466 170011 MOV #FDATIO,R1 ;:SET UP THE LOAD DATA.
498 005470 012701 006420 MOV #FXDAT0,R2
499 005474 012702 006460 MOV #4,R3
500 005500 012703 000004 MOV (R2)+,(R1)+
501 005504 012221 1$: SOB R3,1$
502 005506 077302 MOV #FDATIO,R0 ;:SETUP R0 FOR THE LDD (R0),ACO.
504 005510 012700 006420 MOV #FERR20,ERRVEC ;:IF THE SRC FLOWS FAIL THEN
505 005514 012737 006224 000004 ;: A BUS-ERROR TRAP MAY OCCUR.
506 ;: SET TEST PC FOR OUTPUT.
507 005522 012737 005536 001164 ;: AND SRC POINTER TOO.
508 005530 010037 001166 MOV #F3,@$STMP2
509 005534 005003 CLR R3
510 005536 172410 F3: LDD (R0),ACO
511 005540 005203 F4: INC R3
512 005542 005203 INC R3
513
514 005544 020027 006420 CMP R0,#FDATIO ;:WAS SRC ADDRESS AFFECTED?
515 005550 001417 BEQ 3$ ;: BR IF NOT.
516 005552 010037 001170 MOV R0,@$STMP4 ;: SRC POINTER WAS ALTERED.
517 005556 022700 006410 CMP #FDATIO-10,R0 ;:LOOK LIKE MODE 4?
518 005562 001003 BNE 1$ ;: NO
519 005564 012737 000324 001172 MOV #324,@$STMP5 ;: YES
520 005572 022700 006430 1$: CMP #FDATIO+10,R0 ;:LOOK LIKE MODE 2?
521 005576 001003 BNE 2$ ;: NO
522 005600 012737 000322 001172 MOV #322,@$STMP5 ;: YES
523 005606 104026 2$: ERROR 26 ;: R0 WRONG AFTER LDD.
524
525 005610 004537 004312 3$: JSR R5,CHECK4 ;: WAS SRC DATA AFFECTED ??
526 005614 006460 006420 FXDAT0,FDATIO ;: EXP VS RECD.
527 005620 001407 BEQ 4$ ;: BR IF NOT.
528 005622 012737 006460 001170 MOV #FXDAT0,$STMP4 ;: SOURCE DATA ALTERED...
529 005630 012737 006420 001172 MOV #FDATIO,$STMP5 ;:...BY THE LDD.
530 005636 104025 ERROR 25
531
532 005640 170201 4$: STFPS R1 ;: GET FPS
533 005642 020127 000200 CMP R1,#200 ;: IS IT RIGHT ??
534 005646 001406 BEQ F6 ;: BR IF SO.
535 005650 010137 001166 MOV R1,$STMP3 ;: FPS WRONG...
```

```

536 005654 012737 000200 001170 MOV #200,$TMP4 ;...EXPECTED THIS.
537 005662 104041 ERROR 41
538
539 ; NOW DO THE STORE INSTRUCTION.
540
541 005664 F6: LUPERR ;: LOOF HERE ON ERROR IF SWR9 - 1.
(1) 005664 104411 MOV #4,R4 ;: SET UP THE OUTPUT DATA BUFFER.
542 005666 012704 000004 MOV #FDAT00,R5 ;: SET UP R0 FOR DST MODE 1 REG 0.
543 005672 012705 006440 MOV #-1,(R5)+ ;: IF THE DST FLOWS FAIL
544 005676 012725 177777 1$: SOB R4,1$ ;: A BUS-ERROR TRAP COULD OCCUR
545 005702 077403
546
547 005704 012700 006440 MOV #FDAT00,R0 ;: SET TEST PC.
548 005710 012737 006316 000004 MOV #FERR25,ERRVEC ;: AND DST POINTER.
549
550 005716 012737 005746 001164 MOV #F10,@$TMP2 ;: EXP
551 005724 010037 001166 MOV R0,$TMP3 ;: RECD
552 005730 012737 006460 001170 MOV #FXDAT0,$TMP4
553 005736 012737 006440 001172 MOV #FDAT00,$TMP5
554 005744 005003 CLR R3
555 005746 174010 F10: STD ACO,(R0) ;: TEST INSTRUCTION.
556 005750 005203 F11: INC R3
557 005752 005203 INC R3
558
559 005754 020027 006440 CMP R0,#FDAT00 ;: WAS DST ADDRESS MODIFIED?
560 005760 001417 BEQ 3$ ;: BR IF NOT.
561 005762 010037 001170 MOV R0,@$TMP4 ;: DST POINTER WAS ALTERED.
562 005766 022700 006430 CMP #FDAT00-10,R0 ;: LOOK LIKE DST MODE 4?
563 005772 001003 BNE 1$ ;: NO
564 005774 012737 000644 001172 MOV #644,@$TMP5 ;: YES.
565 006002 022700 006450 1$: CMP #FDAT00+10,R0 ;: LOOK LIKE DST MODE 2?
566 006006 001003 BNE 2$ ;: NO
567 006010 012737 000642 001172 MOV #642,@$TMP5 ;: YES
568 006016 104031 2$: ERROR 31
569
570 006020 012701 006440 3$: MOV #FDAT00,R1
571 006024 012702 006460 MOV #FXDAT0,R2
572 006030 022122 CMP (R1)+,(R2)+ ;: SEE IF THE DATA IS OK.
573 006032 001072 BNE FERR10 ;: BR IF 1ST WORD WRONG.
574 006034 022122 CMP (R1)+,(R2)+
575 006036 001020 BNE FERR6 ;: BR IF 2ND WORD WRONG (BUT GR7 ??)..
576 006040 022122 CMP (R1)+,(R2)+
577 006042 001042 BNE FERR7 ;: BR IF 3RD WORD WRONG (BUT FD ??).
578 006044 022122 CMP (R1)+,(R2)+
579 006046 001064 BNE FERR10 ;: BR IF 4TH WORD WRONG.
580
581 006050 170201 4$: STFPS R1 ;: GET FPS.
582 006052 020127 000200 CMP R1,#200 ;: IS IT RIGHT ??
583 006056 001406 BEQ 5$ ;: BR IF SO.
584 006060 010137 001166 MOV R1,$TMP3 ;: FPS WRONG AFTER STD.
585 006064 012737 000200 001170 MOV #200,$TMP4
586 006072 104041 ERROR 41
587 006074 000137 006470 5$: JMP @#FDONE ;: ALL DONE.
588
589 006100 012701 006442 FERR6: MOV #FDAT01,R1 ;: DID (BUT GR7) FAIL IN THE FDST FLOW ??
590 006104 012702 177777 MOV #-1,R2

```


591	006110	012703	000003		MOV	#3,R3		
592	006114	020221		1\$:	CMP	R2,(R1)-		
593	006116	001003			BNE	2\$; NO	
594	006120	077303			SOB	R3,1\$		
595	006122	104034			ERROR	34	; BUT GR7, FDST FAILED.	
596	006124	000410			BR	4\$		
597								
598	006126	012701	006442	2\$:	MOV	#FDAT01,R1	;DID (BUT GR7) FAIL IN THE FSRC FLOW ?	
599	006132	012703	000003		MOV	#3,R3		
600	006136	005721		3\$:	TST	(R1)+		
601	006140	001027			BNE	FERR10	; NO.	
602	006142	077303			SOB	R3,3\$		
603	006144	104035			ERROR	5	; BUT GR7, FSRC FAILED.	
604	006146	000550		4\$:	BR	FDONE		
605								
606	006150	012701	006444	FERR7:	MOV	#FDAT02,R1	;DID (BUT FD) FAIL IN THE FDST FLOW ?	
607	006154	012702	177777		MOV	#-1,R2		
608	006160	012703	000002		MOV	#2,R3		
609	006164	020221		1\$:	CMP	R2,(R1)+		
610	006166	001003			BNE	2\$; NO	
611	006170	077303			SOB	R3,1\$		
612	006172	104036			ERROR	36	; BUT FD, FDST FAILED.	
613	006174	000410			BR	4\$		
614								
615	006176	012701	006444	2\$:	MOV	#FDAT02,R1	;DID (BUT FD) FAIL IN THE FSRC FLOW ?	
616	006202	012703	000002		MOV	#2,R3		
617	006206	005721		3\$:	TST	(R1)+		
618	006210	001003			BNE	FERR10	; NO	
619	006212	077303			SOB	R3,3\$		
620	006214	104037			ERROR	37	; BUT FD, FSRC FAILED.	
621	006216	000524		4\$:	BR	FDONE		
622								
623	006220	104040		FERR10:	ERROR	40	; DATA INCORRECT.	
624	006222	000522			BR	FDONE		
625								
626	006224	011637	001164	FERR20:	MOV	(SP),@#STMP2	;A FSRC FLOW FAILURE RESULTED	
627	006230	010037	001170		MOV	RO,\$TMP4	; IN A BUS-ERROR ON THE 'LDD'.	
628	006234	021627	005542		CMP	(SP),#F4+2	; LOOK LIKE MODE 6 OR 7 ??	
629	006240	001420			BEQ	3\$; YES.	
630	006242	020027	006416		CMP	RO,#FDAT10-2	; NO, LOOK LIKE MODE 5?	
631	006246	001405			BEQ	1\$; YES.	
632	006250	020027	006422		CMP	RO,#FDAT10+2	; NO, LOOK LIKE MODE 3?	
633	006254	001406			BEQ	2\$; YES.	
634	006256	000137	004224		JMP	@#TRP04	; NO -- WHAT THE HELL ????	
635								
636	006262	012737	000325	001172	1\$:	MOV	#325,@#STMP5	; REPORT MODE 5.
637	006270	000407			BR	4\$		
638	006272	012737	000323	001172	2\$:	MOV	#323,@#STMP5	; REPORT MODE 3.
639	006300	000403			BR	4\$		
640	006302	012737	000326	001172	3\$:	MOV	#326,@#STMP5	; REPORT MODE 6 OR 7.
641	006310	022626			4\$:	CMP	(SP)+,(SP)+	
642	006312	104042			ERROR	42		
643	006314	000406			BR	FDONE		
644								
645	006316	011637	001164	FERR25:	MOV	(SP),@#STMP2	;FDST FLOW FAILURE RESULTED	
646	006322	010037	001170		MOV	RO,\$TMP4	;IN A BUS-ERROR ON THE 'STD'.	

```
647 006326 021627 005752      CMP      (SP),#F11+2      ;LOOK LIKE FDST MODE 6 OR 7?
648 006332 001420      BEQ      3$              ; YES
649 006334 020027 006436      CMP      R0,#FDAT00-2    ;LOOK LIKE MODE 5?
650 006340 001405      BEQ      1$              ; YES
651 006342 020027 006442      CMP      R0,#FDAT00+2    ;LOOK LIKE MODE 3?
652 006346 001406      BEQ      2$              ; YES
653 006350 000137 004224      JMP      @#TRP04         ; NO. ??????????
654
655 006354 012737 000645 001172 1$:      MOV      #645,@#$TMP5    ; REPORT MODE 5.
656 006362 000407      BR      4$
657 006364 012737 000643 001172 2$:      MOV      #643,@#$TMP5    ; REPORT MODE 3
658 006372 000403      BR      4$
659 006374 012737 000646 001172 3$:      MOV      #646,@#$TMP5    ; REPORT MODE 6 OR 7.
660 006402 022626 4$:      CMP      (SP)+,(SP)+
661 006404 104043      ERROR   43
662 006406 000430      BR      FDONE
663
664 006410 177777 177777 177777      .WORD   -1,-1,-1,-1    ; INSURANCE FOR MODES 3 AND 5.
665 006420 177777      FDATA0: -1
666 006422 177777      FDATA1: -1
667 006424 177777      FDATA2: -1
668 006426 177777      FDATA3: -1
669 006430 177777 177777 177777      .WORD   -1,-1,-1,-1
670 006440 177777      FDATA0: -1
671 006442 177777      FDATA1: -1
672 006444 177777      FDATA2: -1
673 006446 177777      FDATA3: -1
674 006450 177777 177777 177777      .WORD   -1,-1,-1,-1
675 006460 052525      FXDATA: 052525
676 006462 031463      FXDAT1: 031463
677 006464 007417      FXDAT2: 007417
678 006466 000477      FXDAT3: 000477
679
680 006470      FDATA:
(1) 006470 104412      CLRFP5
(3) 006472 000400      BR      TST7           ;; CLEAR FP STATUS...
                               ;...AND PROCEED.
```

```
688 (3) *****  
689 (4) *TEST 7 LDD AND LDF -- FSRC MODE 0  
690 (4) *  
691 (4) *THIS IS A TEST OF THE FSRC MODE 0, USING THE INSTRUCTIONS  
692 (4) * LDD AC1,ACO AND LDF AC1,ACO  
693 (3) *****  
694 (2) 006474 000004 TST7: SCOPE  
695 (1) 006476 104411 ; DO THE DOUBLE FIRST (FD=1)  
696 006500 170011 I1: LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
697 006502 012700 007262 SETD ;SET FD.  
698 006506 012701 007232 MOV #IDATIO,R0  
699 006512 012702 000004 MOV #IPATIO,R1  
700 006516 012120 1$: MOV #4,R2  
701 006520 077202 SOB (R1)+,(R0)+ ;SET THE INPUT BUFFER.  
702 006522 012737 007046 000004 MOV #IERR0,@#ERRVEC ;A BUS-ERROR TRAP MAY OCCUR.  
703 006530 012700 007262 MOV #IDATIO,R0  
704 (1) 006534 172510 LDD (R0),AC1 ;: LOAD AC1  
705 006536 012700 007242 MOV #IPATIO,R0  
706 (1) 006542 172410 LDD (R0),ACO ;: LOAD ACO  
707 006544 012737 006574 001164 MOV #I3,@#STMP2  
708 006552 012737 007262 001170 MOV #IDATIO,$TMP4 ; EXP DATA ADDRESS.  
709 006560 012737 007252 001172 MOV #IDATIO,$TMP5 ; RCD  
710 006566 012737 000104 001174 MOV #D,$TMP6 ; FOR ERROR MSG.  
711 006574 172401 I3: LDD AC1,ACO ;TEST INSTRUCTION.  
712 006576 000240 I4: NOP  
713 006600 000240 I5: NOP  
714 006602 012700 007252 MOV #IDATIO,R0  
715 (1) 006606 174010 STD ACO,(R0) ;: STORE ACO  
716 006610 004537 004312 JSR R5,CHECK4 ; CHECK IT  
717 006614 007262 007252 IDATIO,IDATIO  
718 006620 001422 BEQ I11 ; BR IF OK.  
719 006622 012700 007256 MOV #IDATIO2,R0 ;SEE IF (BUT FD) FAILED.  
720 006626 012702 000002 MOV #2,R2  
721 006632 005720 2$: TST (R0)+  
722 006634 001002 BNE 3$  
723 006636 077203 SOB R2,2$  
724 006640 000551 BR IERR2 ; BUT FD FAILED.  
725 006642 012700 007256 3$: MOV #IDATIO2,R0  
726 006646 012702 000002 MOV #2,R2  
727 006652 022720 177777 4$: CMP #-1,(R0)+  
728 006656 001002 BNE 5$  
729 006660 077204 SOB R2,4$  
730 006662 000540 BR IERR2 ; BUT FD FAILED.  
731 006664 000524 5$: BR IERR1 ; DATA INCORRECT.  
732 ;  
733 ;NOW TEST THE LOAD INSTRUCTION WITH FD CLEAR.  
734 ;  
735 I11: LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
736 (1) 006666 104411
```

```
732 006670 012700 007232      MOV      #IPAT10,R0
733 006674 012701 007262      MOV      #IDATIO,R1
734 006700 012702 000004      MOV      #4,R2
735 006704 012021          1$:  MOV      (R0)+,(R1)+
736 006706 077202          SOB      R2,1$
737
738 006710 012700 007262      MOV      #IDATIO,R0
(1) 006714 172510          LDD      (R0),AC1          ;; LOAD AC1
739 006716 012700 007242      MOV      #IPAT20,R0
(1) 006722 172410          LDD      (R0),AC0          ;; LOAD AC0
740 006724 012701 000001      MOV      #1,R1
741 006730 012737 006746 001164  MOV      #I14,@#STMP2
742 006736 012737 000106 001174  MOV      #'F,$TMP6          ; FOR ERROR MSG.
743 006744 170001          SETF          ; CLEAR FD.
744 006746 172401          114:  LDF      AC1,AC0          ; TEST INSTRUCTION.
745 006750 000240          115:  NOP
746 006752 000240          116:  NOP
747 006754 170200          STFPS      R0
748 006756 020027 000004      CMP      R0,#4
749 006762 001111          BNE      IERR4          ; FPS IS WRONG.
750
751 006764 170011          117:  SETD          ; RETURN TO DOUBLE MODE.
752 006766 012700 007252      MOV      #IDATIO0,R0
(1) 006772 174010          STD      AC0,(R0)          ;; STORE AC0
753 006774 012737 177777 007266  MOV      #-1,@#IDATI2
754 007002 012737 177777 007270  MOV      #-1,@#IDATI3
755 007010 004537 004312          JSR      R5,CHECK4          ; CHECK RETURN DATA.
756 007014 007262 007252      IDATIO, IDATIO0
757 007020 001411          BEQ      1$          ; BR IF OK.
758
759 007022 023737 007256 007236  CMP      @#IDATIO2,@#IPAT12      ; DID (BUT FD) FAIL?
760 007030 001042          BNE      IERR1          ; NO
761 007032 023737 007260 007240  CMP      @#IDATIO3,@#IPAT13
762 007040 001036          BNE      IERR1          ; NO
763 007042 000452          BR      IERR3          ; YES, BUT FD FAILED.
764
765 007044 000512          1$:  BR      IDONE          ; NO ERRORS.
766
767          ; ON BUS-ERROR COME HERE TO ANALYZE THE FSRC FAILURE.
768
769 007046 022716 006576          IERR0:  CMP      #14,(SP)          ; MAKE SURE THE TRAP OCCURRED ON
770 007052 001413          BEQ      1$          ; THE INSTRUCTION BEING TESTED.
771 007054 022716 006600          CMP      #15,(SP)
772 007060 001410          BEQ      1$
773 007062 022716 006750          CMP      #115,(SP)
774 007066 001405          BEQ      1$
775 007070 022716 006752          CMP      #116,(SP)
776 007074 001402          BEQ      1$
777 007076 000137 004224          JMP      @#TRP04          ; ???
778 007102 011637 001164          1$:  MOV      (SP),@#STMP2          ; REPORT FAILURE.
779 007106 022626          CMP      (SP)+,(SP)+
780 007110 012737 000627 001166  MOV      #627,@#STMP3
781 007116 012737 000320 001170  MOV      #320,@#STMP4
782 007124 113737 001174 046721  MOV      $TMP6,EM47+2
783 007132 104047          2$:  MOV      ERROR,47          ; 'F' OR 'D' IN TEXT.
784 007134 000456          BR      IDONE          ; SRC MODE 0 TRAPS.
```

```

785
786      :REPORT DATA ERROR.
787
788 007136 012737 007262 001170 IERR1: MOV    #IDATIO,@#STMP4
789 007144 012737 007252 001172      MOV    #IDAT00,@#STMP5
790 007152 113737 001174 047126      MOVB   $TMP6,EM51+2      ; 'F' OR 'D' IN TEXT.
791 007160 104051      1$:      ERROR    51
792 007162 000443      BR      IDONE
793
794      :REPORT FAILURE OF (BUT FD)
795
796 007164 000240 000240 IERR2: 240,240
797 007170 000240 000240 IERR3: 240,240
798 007174 113737 001174 047020      MOVB   $TMP6,EM50+2      ; 'F' OR 'D' IN TEXT.
799 007202 104050      1$:      ERROR    50
800 007204 000432      BR      IDONE
801
802      :REPORT INCORRECT FPS AFTER LOAD INSTRUCTION.
803
804 007206 012737 006746 001164 IERR4: MOV    #I14,@#STMP2
805 007214 010037 001166      MOV    R0,@#STMP3
806 007220 012737 000004 001170      MOV    #4,@#STMP4
807 007226 104041      1$:      ERROR    41
808 007230 000420      BR      IDONE
809
810 007232 000000      IPAT10: 0
811 007234 170360      IPAT11: 170360
812 007236 016161      IPAT12: 016161
813 007240 052525      IPAT13: 052525
814
815 007242 177777      IPAT20: -1
816 007244 177777      IPAT21: -1
817 007246 177777      IPAT22: -1
818 007250 177777      IPAT23: -1
819
820 007252 000000      IDAT00: 0
821 007254 000000      IDAT01: 0
822 007256 000000      IDAT02: 0
823 007260 000000      IDAT03: 0
824
825 007262 000000      IDATIO: 0
826 007264 000000      IDATI1: 0
827 007266 000000      IDATI2: 0
828 007270 000000      IDATI3: 0
829
830 007272      IDONE:
(1) 007272 104412      CLRFPS
(3) 007274 000400      BR      TST10      :: CLEAR FP STATUS...
                        ::...AND PROCEED.

```

```

837      ;*****
(3)      ;*TEST 10      STD AND STF -- FDST MODE 0
(4)      ;*
(4)      ;*THIS IS A TEST OF THE STORE INSTRUCTIONS, STD AND STF, WITH FDST MODE 0.
(4)      ;*
(3)      ;*****
(2) 007276 000004 TST10: SCOPE
838 007300 T1:
(1) 007300 104411      LUPERR      ;; LOOP HERE ON ERROR IF SWR9 = 1.
839 007302 170011      SETD      ;;SET FD
840 007304 012700 010004      MOV      #TPAT10,R0
841 007310 012701 010034      MOV      #TDAT10,R1
842 007314 012702 000004      MOV      #4,R2
843 007320 012021 1$:      MOV      (R0)+,(R1)+ ;SET UP THE INPUT DATA BUFFER.
844 007322 077202      SOB      R2,1$
845
846 007324 012737 007620 000004      MOV      #TERRO,@#ERRVEC ; A BUS-ERROR TRAP MAY RESULT.
847 007332 012700 010034      MOV      #TDAT10,R0
(1) 007336 172410      LDD      (R0),AC0      ;; LOAD AC0
848 007340 012700 010014      MCV      #TPAT20,R0
(1) 007344 172510      LDD      (R0),AC1      ;; LOAD AC1
849 007346 012737 007376 001164      MOV      #T3,@#STMP2
850 007354 012737 010034 001170      MOV      #TDAT10,$TMP4      ; EXP DATA ADDRESS.
851 007362 012737 010024 001172      MOV      #TDAT00,$TMP5      ; RCD DATA ADDRESS.
852 007370 012737 000104 001174      MOV      #'D,$TMP6      ; FOR ERROR MSG.
853 007376 174001      T3:      STD      AC0,AC1
854 007400 000240      T4:      NOP
855 007402 000240      T5:      NOP
856 007404 012700 010024      MOV      #TDAT00,R0
(1) 007410 174110      STD      AC1,(R0)      ;; STORE AC1
857 007412 004537 004312      JSR      R5,CHECK4      ; CHECK IT.
858 007416 010034 010024      TDAT10,TDAT00
859 007422 001410      BEQ      T11      ; BR IF OK.
860
861 007424 012703 010030      MOV      #TDAT02,R3      ;DID (BUT FD) FAIL?
862 007430 012705 000002      MOV      #2,R5
863 007434 005723 1$:      TST      (R3)+
864 007436 001124      BNE      TERR1      ; NO, DATA INCORRECT.
865 007440 077503      SOB      R5,1$
866 007442 000535      BR      TERR2      ; YES, BUT FD FAILED.
867
868      ;NOW TEST THE STF AC0,AC1 INSTRUCTION.
869
870 007444 T11:
(1) 007444 104411      LUPERR      ;; LOOP HERE ON ERROR IF SWR9 = 1.
871 007446 012700 010004      MOV      #TPAT10,R0      ;SET UP THE INPUT DATA BUFFER.
872 007452 012701 010034      MOV      #TDAT10,R1
873 007456 012702 000004      MOV      #4,R2
874 007462 012021 1$:      MOV      (R0)+,(R1)+
875 007464 077202      SOB      R2,1$
876
877 007466 012700 010034      MOV      #TDAT10,R0
(1) 007472 172410      LDD      (R0),AC0      ;; LOAD AC0
878 007474 012700 010014      MOV      #TPAT20,R0
(1) 007500 172510      LDD      (R0),AC1      ;; LOAD AC1
879 007502 012737 007520 001164      MOV      #T14,@#STMP2

```

```
880 007510 012737 000106 001174      MOV    #'F,$STMP6      ; FOR ERROR MSG.
881 007516 170001                    SETF                    ; CLEAR FD
882 007520 174001                    T14:  STF    ACO,AC1
883 007522 000240                    T15:  NOP
884 007524 000240                    T16:  NOP
885 007526 170200                    STFPS    RO
886 007530 020027 000010      CMP    RO,#10
887 007534 001111                    BNE     TERR3          ; FPS IS WRONG.
888
889 007536 170011                    SETD                    ; SET FD.
890 007540 012700 010024      MOV    #TDAT00,RO
(1) 007544 174110                    STD     AC1,(RO)      ;; STORE AC1
891 007546 012737 177777 010040      MOV    #-1,@#TDAT12
892 007554 012737 177777 010042      MOV    #-1,@#TDAT13
893 007562 004537 004312      JSR    R5,CHECK4     ; CHECK DATA.
894 007566 010034 010024      TDATIO,TDAT00
895 007572 001411                    BEQ    T23           ; BR IF OK.
896
897 007574 023737 010030 010010      CMP    @#TDAT02,@#TPAT12 ; DID (BUT FD) FAIL.
898 007602 001042                    BNE     TERR1        ; NO
899 007604 023737 010032 010012      CMP    @#TDAT03,@#TPAT13
900 007612 001036                    BNE     TERR1        ; NO
901 007614 000452                    BR     TERR4        ; YES, BUT FD FAILED.
902
903 007616 000512                    T23:  BR     TDONE
904
905                    ; TRAP HERE THROUGH VECTOR 4 IF A BUS-ERROR OCCURS.
906 007620 022716 007400      TERRO: CMP    #T4,(SP)      ; MAKE SURE THE TRAP WAS ON
907 007624 001413                    BEQ    1$           ; AN INSTRUCTION BEING TESTED.
908 007626 022716 007402      CMP    #T5,(SP)
909 007632 001410                    BEQ    1$
910 007634 022716 007522      CMP    #T15,(SP)
911 007640 001405                    BEQ    1$
912 007642 022716 007524      CMP    #T16,(SP)
913 007646 001402                    BEQ    1$
914 007650 000137 004224      JMP    @#TRP04
915
916 007654 011637 001164      1$:  MOV    (SP),@#STMP2
917 007660 022626                    CMP    (SP)+,(SP)+
918 007662 012737 000527 001166      MOV    #527,@#STMP3
919 007670 012737 000640 001170      MOV    #640,@#STMP4
920 007676 113737 001174 052270      MOV    STMP6,EM121+2 ; 'F' OR 'D' IN TEXT.
921 007704 104121      2$:  ERROR  121
922 007706 000456                    BR     TDONE
923
924                    ; REPORT DATA FAILURE.
925
926 007710 012737 010034 001170      TERR1: MOV    #TDATIO,@#STMP4
927 007716 012737 010024 001172      MOV    #TDAT00,@#STMP5
928 007724 113737 001174 052475      MOV    STMP6,EM123+2 ; 'F' OR 'D' IN TEXT.
929 007732 104123      1$:  ERROR  123
930 007734 000443                    BR     TDONE
931
932                    ; REPORT FAILURE OF (BUT FD).
933
934 007736 000240 000240      TERR2: 240,240
```

```
935 007742 000240 000240 TERR4: 240,240
936 007746 113737 001174 052367 MOVB $TMP6,EM122+2 ; 'F' OR 'D' IN TEXT.
937 007754 104122 1$: ERROR 122
938 007756 000432 BR TDONE
939
940 ;REPORT INCORRECT FPS AFTER STORE INSTRUCTION.
941
942 007760 012737 007522 001164 TERR3: MOV #T15,@$TMP2
943 007766 010037 001166 MOV R0,@$TMP3
944 007772 012737 000010 001170 MOV #10,@$TMP4
945 010000 104041 1$: ERROR 41
946 010002 000420 BR TDONE
947
948 010004 000000 TPAT10: 0
949 010006 170360 TPAT11: 170360
950 010010 016161 TPAT12: 016161
951 010012 052525 TPAT13: 052525
952
953 010014 177777 TPAT20: -1
954 010016 177777 TPAT21: -1
955 010020 177777 TPAT22: -1
956 010022 177777 TPAT23: -1
957
958 010024 000000 TDATA0: 0
959 010026 000000 TDATA1: 0
960 010030 000000 TDATA2: 0
961 010032 000000 TDATA3: 0
962
963 010034 000000 TDATA10: 0
964 010036 000000 TDATA11: 0
965 010040 000000 TDATA12: 0
966 010042 000000 TDATA13: 0
967
968 010044 TDONE:
(1) 010044 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 010046 000400 BR TST11 ;:...AND PROCEED.
```


982
 (3)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2)
 983
 984
 985
 986
 987
 988
 989
 990
 991
 992
 993
 994
 995
 996
 997
 998
 999
 1000
 1001
 1002
 1003
 1004
 1005
 1006
 1007
 1008
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 (1)
 1016
 (1)
 1017
 1018
 1019
 1020
 (1)
 1021

010050 000004
 010052 170011
 010054 012737 010546 001166
 010062 012737 010556 001170
 010070 005003
 010072 004737 010140
 010076 005203
 010100 004737 010140
 010104 005203
 010106 004737 010140
 010112 005203
 010114 004737 010140
 010120 005203
 010122 004737 010140
 010126 005203
 010130 004737 010140
 010134 000137 010566
 010140 005037 010546
 010144 005037 010550
 010150 005037 010552
 010154 012737 000001 010554
 010162 000410
 010164 005137 010546
 010170 005137 010550
 010174 005137 010552
 010200 005137 010554
 010204 012702 000100
 010210 104411
 010212 020327 000000
 010216 001011
 010220 012737 010226 001164
 010226 012700 010546
 010232 172410
 010234 012700 010556
 010240 174010
 010242 020327 000001
 010246 001011
 010250 012737 010256 001164
 010256 012700 010546
 010262 172510
 010264 012700 010556

```

*****
*TEST 11      ACCUMULATORS -- DATA PATTERNS
*
*THIS IS A TEST OF THE FLOATING POINT PROCESSOR ACCUMULATORS.
*EACH ACCUMULATOR IS TESTED IN TWO WAYS:
* 1 TEST PATTERN GENERATED BY FLOATING A ONE ACROSS
*   A FIELD OF ZEROES.
* 2 TEST PATTERN GENERATED BY FLOATING A ZERO ACROSS
*   A FIELD OF ONES.
*EACH OF ACCUMULATORS AC0 THROUGH AC5 IS TESTED.
*NOTE THAT AC0 IS USED TO ACCESS AC4 AND AC5.
*****
TST11: SCOPE
G1:  SETD      ; SET DOUBLE MODE.
      MOV      #GPAT,$TMP3 ; SET EXP...
      MOV      #GDAT,$TMP4 ; ...AND REC POINTERS.
      CLR      R3         ; AC NUMBER UNDER TEST.
      JSR      PC,G2      ; TEST AC0.
      INC      R3
      JSR      PC,G2      ; TEST AC1.
      INC      R3
      JSR      PC,G2      ; TEST AC2.
      INC      R3
      JSR      PC,G2      ; TEST AC3.
      INC      R3
      JSR      PC,G2      ; TEST AC4.
      INC      R3
      JSR      PC,G2      ; TEST AC5.
      JMP      GDONE
G2:  CLR      GPAT        ; GO A FLOATING '1' FIRST...
      CLR      GPAT+2
      CLR      GPAT+4
      MOV      #1,GPAT+6
      BR      G4
G3:  COM      GPAT        ; ...THEN A FLOATING '0'.
      COM      GPAT+2
      COM      GPAT+4
      COM      GPAT+6
G4:  MOV      #64.,R2     ; 64 BIT LOOP CONTROL.
      LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
G5:  CMP      R3,#0
      BNE     1$
      MOV      #.+6,$TMP2
      MOV      #GPAT,R0
      LDD     (R0),AC0 ;: LOAD AC0
      MOV      #GDAT,R0
      STD     AC0,(R0) ;: STORE AC0
1$:  CMP      R3,#1
      BNE     2$
      MOV      #.+6,$TMP2
      MOV      #GPAT,R0
      LDD     (R0),AC1 ;: LOAD AC1
      MOV      #GDAT,R0
  
```

(1)	010270	174110			STD	AC1,(R0)	:: STORE AC1
1022	010272	020327	000002	2\$:	CMP	R3,#2	
1023	010276	001011			BNE	3\$	
1024	010300	012737	010306	001164	MOV	#+6,\$TMP2	
1025	010306	012700	010546		MOV	#GPAT,R0	
(1)	010312	172610			LDD	(R0),AC2	:: LOAD AC2
1026	010314	012700	010556		MOV	#GDAT,R0	
(1)	010320	174210			STD	AC2,(R0)	:: STORE AC2
1027	010322	020327	000003	3\$:	CMP	R3,#3	
1028	010326	001011			BNE	4\$	
1029	010330	012737	010336	001164	MOV	#+6,\$TMP2	
1030	010336	012700	010546		MOV	#GPAT,R0	
(1)	010342	172710			LDD	(R0),AC3	:: LOAD AC3
1031	010344	012700	010556		MOV	#GDAT,R0	
(1)	010350	174310			STD	AC3,(R0)	:: STORE AC3
1032	010352	020327	000004	4\$:	CMP	R3,#4	
1033	010356	001013			BNE	5\$	
1034	010360	012737	010366	001164	MOV	#+6,\$TMP2	
1035	010366	012700	010546		MOV	#GPAT,R0	
(1)	010372	172410			LDD	(R0),AC0	
(1)	010374	174004			STD	AC0,AC4	:: LOAD AC4
1036	010376	012700	010556		MOV	#GDAT,R0	
(1)	010402	172404			LDD	AC4,AC0	
(1)	010404	174010			STD	AC0,(R0)	:: STORE AC4
1037	010406	020327	000005	5\$:	CMP	R3,#5	
1038	010412	001013			BNE	6\$	
1039	010414	012737	010422	001164	MOV	#+6,\$TMP2	
1040	010422	012700	010546		MOV	#GPAT,R0	
(1)	010426	172410			LDD	(R0),AC0	
(1)	010430	174005			STD	AC0,AC5	:: LOAD AC5
1041	010432	012700	010556		MOV	#GDAT,R0	
(1)	010436	172405			LDD	AC5,AC0	
(1)	010440	174010			STD	AC0,(R0)	:: STORE AC5
1042							
1043	010442	004537	004312	6\$:	JSR	R5,CHECK4	: COMPARE...
1044	010446	010546	010556		GPAT,GDAT		:...EXP VS RECD.
1045	010452	001411			BEQ	7\$: BR IF OK.
1046	010454	010300			MOV	R3,R0	
1047	010456	062700	000060		ADD	#'0',R0	
1048	010462	110037	043377		MOVB	R0,EACN	: SET AC NUMBER.
1049	010466	110037	043417		MOVB	R0,RACN	
1050	010472	104044			ERROR	44	: AC BAD.
1051	010474	000420			BR	8\$	
1052							
1053	010476	012700	010546	7\$:	MOV	#GPA1,R0	: NOW ROTATE PATTERN LEFT.
1054	010502	006160	000006		ROL	6(R0)	
1055	010506	006160	000004		ROL	4(R0)	
1056	010512	006160	000002		ROL	2(R0)	
1057	010516	006110			ROL	(R0)	
1058	010520	042760	000001	000006	BIC	#1,6(R0)	
1059	010526	005560	000006		ADC	6(R0)	: BIT 63 TO BIT 0.
1060	010532	005302			DEC	R2	
1061	010534	001226			BNE	G5	: LOOP FOR 64 ROTATES.
1062							
1063	010536	005737	010546	8\$:	TST	GPAT	: FLOATING A '1' ??
1064	010542	001610			BEQ	G3	: YES, GO ROUND FOR A '0'.

1065	010544	000207				RTS	PC		; NO, THIS AC IS DONE, RETURN.
1066									
1067	010546	000000	000000	000000	GPAT:	0,0,0,0			; FLOATING DATA PATTERN.
1068	010556	000000	000000	000000	GDAT:	0,0,0,0			; AC DATA RECEIVER.
1069									
1070	010566				GDONE:				
(1)	010566	104412			CLRFPS				:: CLEAR FP STATUS...
(3)	010570	000400			BR	TST12			::...AND PROCEED.

```

1078
(3)
(4)
(4)
(4)
(4)
(3)
(2) 010572 000004
1079 010574 012727 177777
1080 010600 177777
1081 010602 012700 011222
1082 010606 012701 011342
1083 010612 012703 000024
1084 010616 012120
1085 010620 077302
1086
1087 010622 012700 011222
1088 010626 012701 001166
1089 010632 012702 000012
1090 010636 010021
1091 010640 062700 000010
1092 010644 077204
1093
1094 010646 170011
1095 010650 012700 011222
(1) 010654 172510
1096 010656 012700 011232
(1) 010662 172610
1097 010664 012700 011242
(1) 010670 172710
1098 010672 012700 011252
(1) 010676 172410
(1) 010700 174004
1099 010702 012700 011262
(1) 010706 172410
(1) 010710 174005
1100
1101
1102
1103 010712 012700 011222
1104 010716 010001
1105 010720 012702 000004
1106 010724 005121
1107 010726 172410
1108 010730 174001
1109 010732 004737 011060
1110 010736 077206
1111
1112
1113
1114 010740 012702 000004
1115 010744 010100
1116 010746 005121
1117 010750 172410
1118 010752 174002
1119 010754 004737 011060
  
```

```

*****
*TEST 12 ACCUMULATORS -- DUAL ADDRESSING
*
*THIS TEST PERFORMS A DUAL ADDRESSING TEST ON THE FLOATING ACCUMULATORS.
*NOTE THAT ACCUMULATOR ZERO IS USED TO ACCESS ALL THE OTHERS.
*
*****
TST12: SCOPE
MOV #-1,(PC)+
H1: -1 ; LOOP FLAG.
MOV #HA1W,R0 ; INITIALIZE THE LOAD BUFFER.
MOV #HDAT1,R1
MOV #20,R3
1$: MOV (R1)+,(R0)+
SOB R3,1$

MOV #HA1W,R0
MOV #STMP3,R1
MOV #12,R2
2$: MOV R0,(R1)+ ; SET UP ERROR TEMPS (3 - 14).
ADD #10,R0
SOB R2,2$

H3: SETD ; INIT ALL AC'S.
MOV #HA1W,R0
LDD (R0),AC1 ;; LOAD AC1
MOV #HA2W,R0
LDD (R0),AC2 ;; LOAD AC2
MOV #HA3W,R0
LDD (R0),AC3 ;; LOAD AC3
MOV #HA4W,R0
LDD (R0),AC0 ;; LOAD AC4
STD AC0,AC4 ;; LOAD AC4
MOV #HA5W,R0
LDD (R0),AC0
STD AC0,AC5 ;; LOAD AC5

; NOW COMPLIMENT EACH WORD IN AC1 AND RECHECK ALL AC'S.
MOV #HA1W,R0
MOV R0,R1
MOV #4,R2
1$: COM (R1)+ ; COMPLIMENT A WORD...
LDD (R0),AC0 ;...IN AC1.
STD AC0,AC1 ; CHECK RETURNED DATA.
JSR PC,HSTOR
SOB R2,1$

; DO THE SAME FOR AC2.
MOV #4,R2 ; RESET COUNT...
MOV R1,R0 ;...AND UPDATE SRC POINTER.
2$: COM (R1)+ ; COMPLIMENT...
LDD (R0),AC0
STD AC0,AC2
JSR PC,HSTOR ;...AND CHECK.
  
```

```

1120 010760 077206 SOB R2,2$
1121
1122 : DITTO AC3.
1123 :
1124 010762 012702 000004 MOV #4,R2
1125 010766 010100 MOV R1,R0
1126 010770 005121 3$: COM (R1)+
1127 010772 172410 LDD (R0),AC0
1128 010774 174003 STD AC0,AC3
1129 010776 004737 011060 JSR PC,HSTOR
1130 011002 077206 SOB R2,3$
1131
1132 : DITTO AC4.
1133 :
1134 011004 012702 000004 MOV #4,R2
1135 011010 010100 MOV R1,R0
1136 011012 005121 4$: COM (R1)+
1137 011014 172410 LDD (R0),AC0
1138 011016 174004 STD AC0,AC4
1139 011020 004737 011060 JSR PC,HSTOR
1140 011024 077206 SOB R2,4$
1141
1142 : AND FINALLY, AC5.
1143 :
1144 011026 012702 000004 MOV #4,R2
1145 011032 010100 MOV R1,R0
1146 011034 005121 5$: COM (R1)+
1147 011036 172410 LDD (R0),AC0
1148 011040 174005 STD AC0,AC5
1149 011042 004737 011060 JSR PC,HSTOR
1150 011046 077206 SOB R2,5$
1151
1152 011050 005237 010600 INC H1
1153 011054 001674 BEQ H3 ; GO ROUND ONE MORE TIME...
1154 011056 000555 BR HDONE ;...AND QUIT.
1155
1156 : STORE ALL ACCUMULATORS IN THE OUTPUT BUFFER.
1157 : AND CHECK AGAINST EXPECTED CONTENTS.
1158
1159 011060 011637 011202 HSTOR: MOV (SP),3$ ; COPY RETURN PC.
1160 011064 010016 MOV R0,(SP) ; SAVE R0.
1161 011066 004737 011204 JSR PC,HCLR ; CLEAR ALL OUTPUT BUFFERS.
1162 011072 012700 011272 MOV #HA1R,R0
(1) 011076 174110 STD AC1,(R0) ;; STORE AC1
1163 011100 012700 011302 MOV #HA2R,R0
(1) 011104 174210 STD AC2,(R0) ;; STORE AC2
1164 011106 012700 011312 MOV #HA3R,R0
(1) 011112 174310 STD AC3,(R0) ;; STORE AC3
1165 011114 012700 011322 MOV #HA4R,R0
(1) 011120 172404 LDD AC4,AC0
(1) 011122 174010 STD AC0,(R0) ;; STORE AC4
1166 011124 012700 011332 MOV #HA5R,R0
(1) 011130 172405 LDD AC5,AC0
(1) 011132 174010 STD AC0,(R0) ;; STORE AC5
1167 011134 012600 MOV (SP)+,R0 ; RESTORE R0.
1168 011136 012703 011222 MOV #HA1W,R3

```

1169	011142	012704	011272			MOV	#HA1R,R4		
1170	011146	012705	000024			MOV	#20.,R5		
1171	011152	022324			1\$:	CMP	(R3)+,(R4)+	:	TEST ALL RETURNED AC'S.
1172	011154	001411				BEQ	2\$		
1173	011156	013737	011202	001164		MOV	3\$, \$TMP2		
1174	011164	162737	000010	001164		SUB	#10,\$TMP2	:	SET ERROR PC...
1175	011172	104046				ERROR	46	:	...AND REPORT BAD AC(S).
1176	011174	000401				SKP1			
1177	011176	077513				SOB	R5,1\$		
1178	011200	000137			2\$:	JMP	@(PC)+		
1179	011202	000000			3\$:		0		
1180									
1181									
1182									
1183	011204	012703	011272			HCLR:	MOV	#HA1R,R3	
1184	011210	012704	000024				MOV	#20.,R4	
1185	011214	005023			1\$:	CLR	(R3)+		
1186	011215	077402				SOB	R4,1\$		
1187	011220	000207				RTS	PC		
1188									
1189	011222	000000	000000	000000		HA1W:	.WORD	0,0,0,0	
1190	011232	000000	000000	000000		HA2W:	.WORD	0,0,0,0	
1191	011242	000000	000000	000000		HA3W:	.WORD	0,0,0,0	
1192	011252	000000	000000	000000		HA4W:	.WORD	0,0,0,0	
1193	011262	000000	000000	000000		HA5W:	.WORD	0,0,0,0	
1194									
1195	011272	000000	000000	000000		HA1R:	.WORD	0,0,0,0	
1196	011302	000000	000000	000000		HA2R:	.WORD	0,0,0,0	
1197	011312	000000	000000	000000		HA3R:	.WORD	0,0,0,0	
1198	011322	000000	000000	000000		HA4R:	.WORD	0,0,0,0	
1199	011332	000000	000000	000000		HA5R:	.WORD	0,0,0,0	
1200									
1201	011342	073567	073567	073567		HDAT1:	.WORD	73567,73567,73567,73567	
1202	011352	063146	063146	063146		HDAT2:	.WORD	63146,63146,63146,63146	
1203	011362	010421	010421	010421		HDAT3:	.WORD	10421,10421,10421,10421	
1204	011372	031463	031463	031463		HDAT4:	.WORD	31463,31463,31463,31463	
1205	011402	042104	042104	042104		HDAT5:	.WORD	42104,42104,42104,42104	
1206									
1207	011412					HDONE:			
(1)	011412	104412				CLRFPS			:: CLEAR FP STATUS...
(3)	011414	000400				BR	TST13		::...AND PROCEED.

1216
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 011416 000004
1217
1218
1219
1220 011420 012737 011642 000244
1221 011426 012737 011736 000004
1222 011434 104411
1223 011436 012737 011462 001164
1224 011444 112737 000066 051751
1225 011452 170011
1226 011454 012700 011632
(1) 011460 172410
1227 011462 172406
1228 011464 000241 000241
1229 011470 104111
1230 011472 000240
1231 011474 012737 011520 001164
1232 011502 112737 000067 051751
1233 011510 170011
1234 011512 012700 011632
(1) 011516 172410
1235 011520 172407
1236 011522 000241 000241
1237 011526 104111
1238 011530 000240
1239
1240
1241
1242 011532 104411
1243 011534 012737 011560 001164
1244 011542 112737 000066 052025
1245 011550 170011
1246 011552 012700 011632
(1) 011556 172410
1247 011560 172006
1248 011562 000241 000241
1249 011566 104112
1250 011570 000240
1251 011572 012737 011616 001164
1252 011600 112737 000067 052025
1253 011606 170011
1254 011610 012700 011632
(1) 011614 172410
1255 011616 172007
1256 011620 000241 000241
1257 011624 104112
1258 011626 000240
1259 011630 000452

```
*****  
*TEST 13 LDD AND ADDD -- FSRC MODE 0, WITH ILLEGAL ACCUMULATOR  
*  
*THIS IS A TEST OF FSRC MODE 0 WITH ACCUMULATORS 6 AND 7.  
*ANY ATTEMPT TO ACCESS EITHER AC6 OR AC7 SHOULD CAUSE A TRAP  
*TO VECTOR 244, WITH FEC = 2 (ILLEGAL FPP INSTRUCTION).  
*  
*****  
TST13: SCOPE  
:  
: FIRST THE LOAD CLASS.  
:  
: MOV #STRP1,@#FPVEC ; SET FP VECTOR.  
: MOV #STRP2,@#ERRVEC ; COULD TRAP TO 4 TOO (NON-EX MEM).  
: LUPERR ; LOOP HERE ON ERROR IF SWR9 = 1.  
: MOV #S1,$TMP2 ; SET ERROR PC.  
: MOVB #'6,EM111X ; AND TEXT.  
: SETD ; DOUBLE MODE.  
: MOV #SDAT,R0  
: LDD (R0),ACO ;: LOAD ACO  
S1: LDD AC6,ACO ; ACCESS AC6 USING A LDD.  
: 241,241 ; WE'RE HERE IF IT DIDN'T TRAP !!!  
: ERROR 111  
: NOP  
: MOV #S2,$TMP2 ; CHANGE ERROR PC.  
: MOVB #'7,EM111X ; AND TEXT.  
: SETD  
: MOV #SDAT,R0  
S2: LDD (R0),ACO ;: LOAD ACO  
: LDD AC7,ACO ; AGAIN WITH AC7.  
: 241,241  
: ERROR 111  
: NOP  
:  
: NOW THE NOT-LOAD CLASS USING ADDD.  
:  
: LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
: MOV #S3,$TMP2 ; CHANGE ERROR PC.  
: MOVB #'6,EM112X ; AND TEXT.  
: SETD  
: MOV #SDAT,R0  
: LDD (R0),ACO ;: LOAD ACO  
S3: ADDD AC6,ACO ; ACCESS AC6 USING ADDD.  
: 241,241  
: ERROR 112  
: NOP  
: MOV #S4,$TMP2  
: MOVB #'7,EM112X  
: SETD  
: MOV #SDAT,R0  
S4: LDD (R0),ACO ;: LOAD ACO  
: ADDD AC7,ACO ; AGAIN WITH AC7.  
: 241,241  
: ERROR 112  
: NOP  
: BR SDONE
```

```

1260
1261 011632 010421 021042 031463 SDAT: 10421,21042,31463,42104 ; PRESET VALUE FOR ACO (POS NON-ZERO).
1262
1263 ; WE'RE HERE ON ANY TRAP THRU THE FP VECTOR (244).
1264
1265 011642 027627 000000 000241 STRP1: CMP @ (SP),#241 ; TRAP ON OUR TEST OPCODE ???
1266 011650 001402 BEQ 1$ ; BR IF SO.
1267 011652 000137 004200 JMP @#TRP244 ; ??? WHAT-THE-HELL ???
1268 011656 170204 1$: STFPS R4 ; GET FPS.
1269 011660 020427 100200 CMP R4,#100200 ; FPS CORRECT ??
1270 011664 001406 BEQ 2$
1271 011666 012737 100200 001170 MOV #100200,$TMP4 ; EXPD...
1272 011674 010437 001166 MOV R4,$TMP3 ; ...AND OBSERVED FPS.
1273 011700 104113 ERROR 113
1274 011702 170304 2$: STST R4 ; GET FEC.
1275 011704 020427 000002 CMP R4,#2 ; FEC RIGHT ??
1276 011710 001406 BEQ 3$
1277 011712 012737 000002 001170 MOV #2,$TMP4
1278 011720 010437 001166 MOV R4,$TMP3
1279 011724 104114 ERROR 114
1280 011726 012604 3$: MOV (SP)+,R4 ; GET RETURN PC...
1281 011730 005726 TST (SP)+ ; ...POP THE STACK.
1282 011732 000164 000006 JMP 6(R4) ; ...AND CONTINUE.
1283
1284 ; FAILURE IN (BUT FSRC) TRAPPED TO 4 INSTEAD OF 244.
1285
1286 011736 027627 000000 000241 STRP2: CMP @ (SP),#241 ; TRAP ON INSTRUCTION UNDER TEST ???
1287 011744 001402 BEQ 1$
1288 011746 000137 004224 JMP @#TRP04 ; NO, UNEXPECTED TRAP TO 4 !!!
1289 011752 104115 1$: ERROR 115
1290 011754 000400 BR SDONE
1291
1292 SDONE:
(1) 011756 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 011760 000400 BR TST14 ;: ...AND PROCEED.

```



```
1299          :*****
(3)          :*TEST 14      LDD -- FSRC MODE 2 (AUTO-INCREMENT)
(4)          :*
(4)          :* THIS IS A TEST OF FSRC MODE 2, AUTO-INCREMENT MODE.
(4)          :*
(3)          :*****
(2) 011762 000004 TST14: SCOPE
1300 011764 J1:
(1) 011764 104411      LUPERR      ;; LOOP HERE ON ERROR IF SWR9 1.
1301 011766 012737 012106 000004      MOV      #JERR0,@#ERRVEC
1302 011774 170011      SETD      ;SET DOUBLE MODE
1303 011776 012700 012204      MOV      #JDAT0,R0
(1) 012002 172410      LDD      (R0),AC0      ;; LOAD AC0
1304 012004 012737 012022 001164      MOV      #J2,$TMP2
1305 012012 012700 012164      MOV      #JDAT10,R0
1306 012016 010037 001166      MOV      R0,$TMP3      ; R0 BEFORE EXECUTION.
1307 012022 172420      J2:      LDD      (R0)+,AC0      ;TEST INSTRUCTION
1308 012024 000240      J3:      NOP
1309 012026 000240      J4:      NOP
1310 012030 010037 001170      MOV      R0,$TMP4      ; R0 AFTER.
1311 012034 012700 012174      MOV      #JDAT00,R0
(1) 012040 174010      STD      AC0,(R0)      ;; STORE AC0
1312
1313 012042 023727 001170 012174      CMP      $TMP4,#JDAT10+10 ; DID IT AUTO-INCR ??
1314 012050 001401      BEQ      1$      ; YES.
1315 012052 104053      ERROR   53      ; NO, ERROR.
1316
1317 012054 004537 004312      1$:      JSR      R5,CHECK4      ; DATA OK ??
1318 012060 012164 012174      JDAT10,JDAT00
1319 012064 001407      BEQ      J7      ; YUP.
1320 012066 012737 012164 001170      MOV      #JDAT10,$TMP4
1321 012074 012737 012174 001172      MOV      #JDAT00,$TMP5
1322 012102 104054      ERROR   54      ; DATA INCORRECT.
1323
1324 012104 000443      J7:      BR      JDONE
1325
1326          :IF A TRAP THROUGH 4 OCCURS COME HERE
1327
1328 012106 021627 012024      JERR0:  CMP      (SP),#J3      ;SEE IF THE TRAP
1329 012112 001405      BEQ      J10      ;OCCURRED ON THE
1330 012114 021627 012026      CMP      (SP),#J4      ;TESTED INSTRUCTION
1331 012120 001402      BEQ      J10
1332 012122 000137 004224      JMP      @#TRP04
1333
1334 012126 012737 000762 001172      J10:    MOV      #762,@#$TMP5      ;REPORT FSRC FLOW
1335 012134 012737 000322 001174      MOV      #322,@#$TMP6      ;FAILURE
1336 012142 011637 001164      MOV      (SP),@#$TMP2
1337 012146 022626      CMP      (SP)+,(SP)+
1338 012150 104052      1$:      ERROR   52
1339 012152 000420      BR      JDONE
1340
1341 012154 010421 021042 042104 031463      JBUF0:  010421,021042,042104,031463
1342
1343 012164 052525 114631 063146 073567      JDAT10: 052525,114631,063146,073567
1344
1345 012174 000000 000000 000000 000000      JDAT00: 0,0,0,0
```

CJFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJFPAA.P11 28-JAN-81 09:50 T14

F 5
MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-27
LDD -- FSRC MODE 2 (AUTO-INCREMENT)

1346
1347 012204 177777 177777 177777 JDAT0: -1,-1,-1,-1

1348
1349 012214 JDONE:

(1) 012214 104412
(3) 012216 000400

CLRFPS
BR TST15

:: CLEAR FP STATUS...
::...AND PROCEED.

```

*****
*TEST 15      LDD -- FSRC MODE 4 (AUTO-DECREMENT)
*
* THIS IS A TEST OF FSRC MODE 4, AUTO-DECREMENT MODE.
*
*****
TST15: SCOPE
K1:
LUPERR                ;; LOOP HERE ON ERROR IF SWR9 - 1.
MOV      #KERR0,@#ERRVEC
SETD                    ;SET DOUBLE MODE
MOV      #KPAT0,R0
LDD      (R0),AC0      ;; LOAD AC0
MOV      #K2,$TMP2
MOV      #KBUF0,R0
MOV      R0,$TMP3      ; R0 BEFORE.
LDD      -(R0),AC0    ;TEST INSTRUCTION
K2:
K3:  NOP
K4:  NOP
MOV      R0,$TMP4      ; R0 AFTER.

MOV      #KDAT00,R0
STD      AC0,(R0)      ;; STORE AC0
CMP      $TMP4,#KBUF0-10 ;; DID IT AUTO-DECR ??
BEQ      1$            ; YES.
ERROR    56            ; NO

1$:  JSR      R5,CHECK4 ; DATA CORRECT ??
      KDAT10,KDAT00
BEQ      K7            ; YES.
MOV      #KDAT10,$TMP4
MOV      #KDAT00,$TMP5
ERROR    57            ; DATA WRONG.

K7:  BR      KDONE

; TRAP HERE ON A BUS-ERROR.
KERR0: CMP      (SP),#K3 ;SEE IF THE ERROR
BEQ      K10          ;OCCURRED AT THE
CMP      (SP),#K4    ;INSTRUCTION TESTED.
BEQ      K10
JMP      @#TRP04

K10: MOV      #762,@#$TMP5 ;REPORT FAILURE IN
MOV      #326,@#$TMP4 ;FSRC FLOWS
MOV      (SP),@#$TMP2
1$:  ERROR    55
BR      KDONE

KDAT10: .WORD    052525,114631,063140,073567
KBUF0:  010421,031463,042104,021042
KDAT00:  0,0,0,0
  
```

CJFFAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJFPAA.P11 28-JAN-81 09:50 T15

MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-29
LDD -- FSRC MODE 4 (AUTO-DECREMENT)

SEQ 0059

1403 012440 177777 177777 177777 KPATO:
1404
1405 012450 KDONE:
(1) 012450 104412
(3) 012452 000400

-1,-1,-1,-1

CLRFPS
BR TST16

::: CLEAR FP STATUS...
::: ...AND PROCEED.

```

1413 .....
(3) *TEST 16 LDF -- FSRC MODE 2 (AUTO-INCREMENT)
(4) *
(4) * THIS IS A TEST OF FSRC MODE 2 (AUTO-INCREMENT) IN
(4) * SINGLE PRECISION MODE (FD=0).
(3) .....
(2) 012454 000004 TST16: SCOPE
1414 012456 L1:
(1) 012456 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
1415 012460 170011 SETD ;; SET DOUBLE MODE
1416 012462 012700 012670 MOV #LDATIO,R0 ;; SET UP THE INPUT BUFFER.
1417 012466 012701 012660 MOV #LPAT20,R1
1418 012472 012702 000004 MOV #4,R2
1419 012476 012120 1$: MOV (R1)+,(R0)+
1420 012500 077202 SOB R2,1$
1421
1422 012502 012700 012650 MOV #LPAT10,R0
(1) 012506 172410 LDD (R0),AC0 ;; LOAD AC0
1423 012510 012737 012530 001164 MOV #L2,$TMP2
1424 012516 012700 012670 MOV #LDATIO,R0
1425 012522 010037 001166 MOV R0,$TMP3 ; R0 BEFORE.
1426 012526 170001 SETF
1427 012530 172420 L2: LDF (R0)+,AC0
1428 012532 000240 L3: NOP
1429 012534 000240 NOP
1430 012536 010037 001170 MOV R0,$TMP4 ; R0 AFTER.
1431 012542 170011 SETD ; SET FD
1432 012544 012700 012700 MOV #LDATIO,R0
(1) 012550 174010 STD AC0,(R0) ;; STORE AC0
1433 012552 023727 001170 012674 CMP $TMP4,#LDATIO+4 ; DID IT AUTO-INCR ??
1434 012560 001401 BEQ 1$ ; YES.
1435 012562 104060 ERROR 60 ; NO.
1436
1437 012564 012737 177777 012674 1$: MOV #-1,@#LDATIO+4
1438 012572 012737 177777 012676 MOV #-1,@#LDATIO+6
1439 012600 004537 004312 JSR R5,CHECK4 ; DATA OK ??
1440 012604 012670 012700 LDATIO,LDATIO
1441 012610 001416 BEQ L6 ; YES.
1442 012612 012737 012670 001170 MOV #LDATIO,$TMP4
1443 012620 012737 012700 001172 MOV #LDATIO,$TMP5
1444 012626 004537 004312 JSR R5,CHECK4 ; DID BUT FD FAIL ??
1445 012632 012660 012700 LPAT20,LDATIO
1446 012636 001402 BEQ 2$ ; YES.
1447 012640 104061 ERROR 61 ; NO, DATA WRONG.
1448 012642 000401 SKP1
1449 012644 104062 2$: ERROR 62 ; BUT FD FAILED.
1450
1451 012646 000420 L6: BR LDONE
1452
1453 012650 177777 177777 177777 LPAT10: .WORD -1,-1,-1,-1
1454
1455 012660 052525 114631 063142 LPAT20: 052525,114631,063142,073567
1456
1457 012670 000000 000000 000000 LDATIO: 0,0,0,0
1458

```

CLFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CLFPAA.P11 28-JAN-81 09:50 T16

J 5
MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-31
LDF -- FSRC MODE 2 (AUTO-INCREMENT)

SEQ 0061

1459 012700 000000 000000 000000 LDAT00: 0.0.0.0

1460

1461 012710

(1) 012710 104412

(3) 012712 000400

LDONE:

CLRFPS

BR TST17

:: CLEAR FP STATUS...
::...AND PROCEED.

```

1469          ;*****
(3)          ;*TEST 17          LDD -- FSRC MODE 2 WITH GR7 (PC IMMEDIATE)
(4)          ;*
(4)          ;* THIS IS A TEST OF FSRC MODE 2 USING GR7 (PC).
(4)          ;* THIS IS THE 'IMMEDIATE' MODE.
(4)          ;*
(3)          ;*****
(2) 012714 000004 TST17: SCOPE
1470 012716 M1:
(1) 012716 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
1471 012720 012737 013062 000004 MOV #MERR3,@#ERRVEC
1472 012726 170011 SETD
1473 012730 012700 013122 MOV #MPAT10,R0
(1) 012734 172410 LDD (R0),AC0 ;: LOAD ACO
1474 012736 012737 012750 001164 MOV #M2,$TMP2
1475 012744 012704 012762 MOV #M2+12,R4 ; TO CALCULATE RETURN PC.
1476 012750 172427 M2: LDD (PC)+,AC0 ;TEST INSTRUCTION
1477 012752 005744 TST -(R4) ; 5744 (SOULD SKIP THIS ONE)...
1478 012754 005744 TST -(R4) ;...AND EXECUTE NEXT 3.
1479 012756 005744 TST -(R4)
1480 012760 005744 TST -(R4)
1481 012762 020427 012754 CMP R4,#M2+4 ; RETURN TO RIGHT PC ??
1482 012766 001406 BEQ 1$ ; BR IF SO.
1483 012770 010437 001166 MOV R4,$TMP3
1484 012774 012737 012754 001170 MOV #M2+4,$TMP4
1485 013002 104064 ERROR 64 ; PC WRONG.
1486
1487 013004 1$:
(1) 013004 012700 013142 MOV #MDAT00,R0
(1) 013010 174010 STD AC0,(R0) ;: STORE ACO
1488 013012 004537 004312 JSR R5,CHECK4 ; DATA RIGHT ??
1489 013016 013132 013142 MPAT20,MDAT00
1490 013022 001416 BEQ 3$ ; YES.
1491 013024 012737 013132 001170 MOV #MPAT20,$TMP4
1492 013032 012737 013142 001172 MOV #MDAT00,$TMP5
1493 013040 004537 004312 JSR R5,CHECK4 ; DID BUT GR7 FAIL ??
1494 013044 012752 013142 M2+2,MDAT00
1495 013050 001402 BEQ 2$ ; YES.
1496 013052 104066 ERROR 66 ; NO, DATA INCORRECT.
1497 013054 000401 SKP1
1498 013056 104063 2$: ERROR 63 ; BUT GR7 FAILED.
1499
1500 013060 000434 3$: BR MDONE
1501
1502 ;TRAP TO HERE THROUGH 4.
1503
1504 013062 021627 012752 MERR3: CMP (SP),#M2+2 ; TRAP ON OUR TEST ??
1505 013066 001405 BEQ 1$
1506 013070 021627 012754 CMP (SP),#M2+4
1507 013074 001402 BEQ 1$
1508 013076 000137 004224 JMP TRP04 ; NO, UNFORSEEN TRAP ???
1509
1510 013102 011637 001166 1$: MOV (SP),@#$TMP3 ;IF YES REPORT
1511 013106 012737 012754 001170 MOV #M2+4,@#$TMP4 ;BAD CONSTANT
1512 013114 022626 CMP (SP)+,(SP)+
1513 013116 104065 2$: ERROR 65

```

CJFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJFPAA.P11 28-JAN-81 09:50 T17

MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-33
LDD -- FSRC MODE 2 WITH GR7 (PC IMMEDIATE)

SEQ 0063

1514 013120 000414 BR MDONE
1515
1516 013122 177777 177777 177777 MPAT10: -1,-1,-1,-1
1517
1518 013132 005744 000000 000000 MPAT20: 5744,0,0,0
1519
1520 013142 000000 000000 000000 MDAT00: 0,0,0,0
1521
1522 013152 MDONE:
(1) 013152 104412 CLRFP
(3) 013154 000400 BR TST20

:: CLEAR FP STATUS...
::...AND PROCEED.


```

1529          *****
(3)          *TEST 20          LDD -- FSRC MODE 3 (AUTO-INCR DEFERRED)
(4)          *
(4)          * THIS IS A TEST OF FSRC MODE 3, AUTO INCREMENT DEFERRED.
(4)          *
(3)          *****
(2) 013156 000004          TST20: SCOPE
1530 013160          N1:
(1) 013160 104411          LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
1531 013162 012737 013354 000004          MOV          #NERR0,@#ERRVEC ; IN CASE OF BUS-ERROR.
1532 013170 170011          SETD          ; SET FD MODE
1533 013172 012737 013570 001170          MOV          #NDATIO,$TMP4 ; EXP DATA
1534 013200 012737 013526 001172          MOV          #NDAT00,$TMP5 ; RCD DATA.
1535 013206 012737 013232 001164          MOV          #N2,$TMP2
1536 013214 012700 013560          MOV          #NPAT10,R0
(1) 013220 172410          LDD          (R0),AC0          ;; LOAD ACO
1537 013222 012700 013546          MOV          #NPAT20,R0
1538 013226 010037 001166          MOV          R0,$TMP3          ; SAVE R0 BEFORE.
1539 013232 172430          N2: LDD          @(R0)+,AC0          ; TEST INSTRUCTION.
1540 013234 005203          N3: INC          R3
1541 013236 005203          N4: INC          R3
1542 013240 010046          MOV          R0,-(SP)          ; SAVE R0 AFTER.
1543 013242 012700 013526          MOV          #NDAT00,R0
(1) 013246 174010          STD          ACO,(R0)          ;; STORE ACO
1544 013250 012600          MOV          (SP)+,R0          ; AND RECALL R0.
1545
1546 013252 020027 013550          CMP          R0,#NPAT20+2          ; DID IT AUTO-INCREMENT RIGHT ??
1547 013256 001007          BNE          N5          ; NO, CHECK IJ OUT.
1548 013260 004537 004312          JSR          R5,CHECK4          ; DATA RIGHT ??
1549 013264 013570 013526          NDATIO,NDAT00
1550 013270 001401          BEQ          1$          ; BR IF SO.
1551 013272 104073          ERROR       73          ; DATA WRONG.
1552 013274 000541          1$: BR          NDONE
1553
1554 013276 020027 013556          N5: CMP          R0,#NPAT20+10          ; DID IT DO MODE 2 ?
1555 013302 001462          BEQ          NERR1          ; YES.
1556 013304 020027 013536          CMP          R0,#NPAT20-10          ; DID IT DO MODE 4 ?
1557 013310 001463          BEQ          NERR2          ; YES.
1558 013312 020027 013546          CMP          R0,#NPAT20          ; MODE 0 OR 1 ??
1559 013316 001012          BNE          1$          ; NO ???
1560
1561 013320 004537 004312          JSR          R5,CHECK4          ; DID IT DO MODE 0 ??
1562 013324 013560 013526          NPAT10,NDAT00
1563 013330 001457          BEQ          NERR3          ; YES.
1564 013332 004537 004312          JSR          R5,CHECK4          ; DID IT DO MODE 1 ??
1565 013336 013546 013526          NPAT20,NDAT00
1566 013342 001456          BEQ          NERR4          ; YES.
1567 013344 010037 001170          1$: MOV          R0,$TMP4          ; NONE OF THE ABOVE.
1568 013350 104072          ERROR       72          ; R0 WRONG, DON'T KNOW WHY !
1569 013352 000512          BR          NDONE
1570
1571          ;
1572          ; IF A BUS-ERROR TRAP OCCURS COME HERE
1573          ; TO SEE IF THE FAILURE WAS IN THE FSRC FLOW.
1574 013354 022716 013236          NERR0: CMP          #N4,(SP)          ; FSRC MODE 6 OR ??
1575 013360 001410          BEQ          NERR10          ; YES.

```

```
1576 013362 022716 013234      CMP      #N3,(SP)
1577 013366 001003      BNE      2$
1578 013370 020027 013544      1$:     CMP      R0,#NPAT20-2      ;FSRC MODE 5?
1579 013374 001407      BEQ      NERR11      ; YES.
1580 013376 000137 004224      2$:     JMP      @NTRP04
1581
1582 013402 011637 001164      NERR10: MOV      (SP),@N$TMP2      ;WENT TO MODE 6 OR 7.
1583 013406 022626      CMP      (SP)+,(SP)+
1584 013410 104067      1$:     ERROR   67
1585 013412 000472      BR       N$DONE
1586
1587 013414 011637 001164      NERR11: MOV      (SP),@N$TMP2      ;WENT TO MODE 5
1588 013420 022626      CMP      (SP)+,(SP)+
1589 013422 012737 000627 001174      MOV      #627,$TMP6
1590 013430 012737 000323 001200      MOV      #323,$TMP10
1591 013436 012737 000325 001176      MOV      #325,@N$TMP7
1592 013444 104070      1$:     ERROR   70
1593 013446 000454      BR       N$DONE
1594
1595 013450 012737 000322 001176      NERR1:   MOV      #322,@N$TMP7      ;FSRC MODE 2.
1596 013456 000403      SKP3
1597 013460 012737 000324 001176      NERR2:   MOV      #324,@N$TMP7      ;FSRC MODE 4
1598 013466 000403      SKP3
1599 013470 012737 000320 001176      NERR3:   MOV      #320,@N$TMP7      ;FSRC MODE 0
1600 013476 000403      SKP3
1601 013500 012737 000321 001176      NERR4:   MOV      #321,@N$TMP7      ;FSRC MODE 1
1602 013506 012737 000627 001174      MOV      #627,@N$TMP6
1603 013514 012737 000323 001200      MOV      #323,@N$TMP10
1604 013522 104071      1$:     ERROR   71
1605 013524 000425      BR       N$DONE
1606
1607 013526 000000 000000 000000      NDATA0: 0,0,0,0
1608
1609 013536 052525 052525 052525      52525,52525,52525,52525
1610
1611 013546 013570 070707 070707      NPAT20:  NDATA10,070707,070707,070707
1612 013556 000001      .WORD   1
1613 013560 177777 177777 177777      NPAT10: -1,-1,-1,-1
1614
1615 013570 010421 021042 031463      NDATA10: 010421,021042,031463,042104
1616
1617 013600      N$DONE:
(1) 013600 104412      CLRFP5      ;; CLEAR FP STATUS...
(3) 013602 000403      BR          TST21      ;;...AND PROCEED.
```

```
1624 .....  
(3) *TEST 21 LDD -- FSRC MODE 5 (AUTO-DECR DEFERRED)  
(4) *  
(4) * THIS IS A TEST OF FSRC MODE 5, AUTO DECREMENT DEFERRED.  
(4) *  
(3) .....  
(2) 013604 000004 TST21: SCOPE  
1625 013606 01: LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
(1) 013606 104411 MOV #OERR0,ERRVEC ;: SET BUS-ERROR TRAP.  
1626 013610 012737 014002 000004 SETD ;: SET FD MODE  
1627 013616 170011 MOV #O2,$TMP2  
1628 013620 012737 013660 001164 MOV #ODATIO,$TMP4  
1629 013626 012737 014216 001170 MOV #ODAT00,$TMP5  
1630 013634 012737 014154 001172 MOV #OPAT10,R0  
1631 013642 012700 014206 LDD (R0),AC0 ;: LOAD AC0  
(1) 013646 172410 MOV #OPAT21,R0 ;: R0 BEFORE.  
1632 013650 012700 014174 MOV R0,$TMP3 ;: TEST INSTRUCTION  
1633 013654 010037 001166 02: LDD @-(R0),AC0  
1634 013660 172450 03: INC R3  
1635 013662 005203 04: INC R3  
1636 013664 005203 MOV R0,-(SP) ;: SAVE FINAL R0.  
1637 013666 010046 MOV #ODAT00,R0  
1638 013670 012700 014154 STD AC0,(R0) ;: STORE AC0  
(1) 013674 174010 MOV (SP)+,R0 ;: AND RECALL R0.  
1639 013676 012600  
1640  
1641 013700 020027 014172 CMP R0,#OPAT20 ;: WAS R0 DECREMENTED BY 2 ??  
1642 013704 001007 BNE O5 ;: NO, CHECK IT OUT.  
1643 013706 004537 004312 JSR R5,CHECK4 ;: YES, IS DATA RIGHT ??  
1644 013712 014216 014154 ODAT10,ODAT00  
1645 013716 001401 BEQ 1$ ;: BR IF SO.  
1646 013720 104100 ERROR 100 ;: DATA INCORRECT.  
1647 013722 000541 1$: BR ODONE  
1648  
1649 013724 020027 014204 05: CMP R0,#OPAT21+10 ;: FSRC MODE 2 ??  
1650 013730 001462 BEQ OERR1 ;: YES.  
1651 013732 020027 014164 CMP R0,#OPAT21-10 ;: FSRC MODE 4?  
1652 013736 001463 BEQ OERR2 ;: YES.  
1653 013740 020027 014174 CMP R0,#OPAT21 ;: MODE 0 OR 1 ??  
1654 013744 001012 BNE 1$ ;: NO ???  
1655  
1656 013746 004537 004312 JSR R5,CHECK4 ;: IS IT MODE 0 ??  
1657 013752 014206 014154 OPAT10,ODAT00  
1658 013756 001457 BEQ OERR3 ;: YES.  
1659 013760 004537 004312 JSR R5,CHECK4 ;: IS IT MODE 1 ??  
1660 013764 014174 014154 OPAT21,ODAT00  
1661 013770 001456 BEQ OERR4 ;: YES.  
1662  
1663 013772 010037 001170 1$: MOV R0,$TMP4 ;: NONE OF THE ABOVE...  
1664 013776 104077 ERROR 77 ;: R0 WRONG.  
1665 014000 000512 BR ODONE  
1666  
1667 ;: IF A BUS-ERROR TRAP OCCURS COME  
1668 ;: HERE TO SEE IF THE FAILURE WAS IN THE FSRC FLOW.  
1669  
1670 014002 022716 013664 OERR0: CMP #O4,(SP) ;: FSRC MODE 6 OR 7?
```

```

1671 014006 001410 BEQ OERR10 ; YES.
1672 014010 022716 013662 CMP #03,(SP)
1673 014014 001003 BNE 2$
1674 014016 020027 014176 1$: CMP R0,#OPAT21+2 ;FSRC MODE 3?
1675 014022 001407 BEQ OERR11 ; YES.
1676 014024 000137 004224 2$: JMP @#TRP04
1677
1678 014030 011637 001164 OERR10: MOV (SP),@#STMP2 ;MODE 6 OR 7
1679 014034 022626 CMP (SP)+,(SP)+
1680 014036 104074 1$: ERROR 74
1681 014040 000472 BR ODONE
1682
1683 014042 011637 001164 OERR11: MOV (SP),@#STMP2 ;WENT TO FSRC MODE 3
1684 014046 022626 CMP (SP)+,(SP)+
1685 014050 012737 000627 001176 MOV #627,@#STMP7
1686 014056 012737 000325 001200 MOV #325,@#STMP10
1687 014064 012737 000323 001174 MOV #323,@#STMP6
1688 014072 104075 1$: ERROR 75
1689 014074 000454 BR ODONE
1690
1691 014076 012737 000322 001174 OERR1: MOV #322,@#STMP6 ;FSRC MODE2
1692 014104 000403 SKP3
1693 014106 012737 000324 001174 OERR2: MOV #324,@#STMP6 ;FSRC MODE 4
1694 014114 000403 SKP3
1695 014116 012737 000320 001174 OERR3: MOV #320,@#STMP6 ;FSRC MODE 0
1696 014124 000403 SKP3
1697 014126 012737 000321 001174 OERR4: MOV #321,@#STMP6 ;FSRC MODE 1
1698 014134 012737 000627 001176 MOV #627,@#STMP7
1699 014142 012737 000325 001200 MOV #325,@#STMP10
1700 014150 104076 1$: ERROR 76
1701 014152 000425 BR ODONE
1702
1703 014154 000000 000000 000000 ODAT00: .WORD 0,0,0,0
1704 014164 052525 052525 052525 .WORD 52525,52525,52525
1705
1706 014172 014216 OPAT20: .WORD ODAT10
1707 014174 070707 070707 070707 OPAT21: .WORD 070707,070707,070707,070707
1708 014204 000001 .WORD 1
1709 014206 177777 177777 177777 OPAT10: .WORD -1,-1,-1,-1
1710
1711 014216 073567 004210 114631 ODAT10: .WORD 073567,004210,114631,125252
1712
1713 014226 ODONE:
(1) 014226 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 014230 000400 BR TST2 ;:...AND PROCEED.

```

```

1720          ::*****
(3)          ::*TEST 22      LDD -- FSRC MODE 6 (INDEX)
(4)          ::*
(4)          ::* THIS IS A TEST OF FSRC MODE 6, INDEX MODE
(4)          ::*
(3)          ::*****
(2) 014232 000004 TST22: SCOPE
1721 014234          P1:
(1) 014234 104411          LUPERR          ;; LOOP HERE ON ERROR IF SWR9 1.
1722 014236 012737 014402 000004          MOV      #PERRO,ERRVEC  ;; SET BUS-ERROR TRAP.
1723 014244 012737 014306 001164          MOV      #P2,$TMP2
1724 014252 012737 014574 001170          MOV      #PDATIO,$TMP4
1725 014260 012737 014604 001172          MOV      #PDAT00,$TMP5
1726 014266 170011          SETD          ;SET FD MODE
1727 014270 012700 014564          MOV      #PPAT10,R0
(1) 014274 172410          LDD      (R0),AC0      ;; LOAD AC0
1728 014276 012700 014333          MOV      #PDATIO-241,R0
1729 014302 010037 001166          MOV      R0,$TMP3
1730 014306 172460 000241          P2: LDD      241(R0),AC0  ; TEST INSTRUCTION.
1731          P3-P2+2
1732 014312 000241          P4: CLC
1733 014314 000241          CLC
1734 014316 010046          MOV      R0,-(SP)      ; SAVE R0 AFTER.
1735 014320 012700 014604          MOV      #PDAT00,R0
(1) 014324 174010          STD      AC0,(R0)      ;; STORE AC0
1736 014326 012600          MOV      (SP)+,R0      ; AND RECALL R0.
1737
1738 014330 020027 014333          CMP      R0,#PDATIO-241 ; R0 CORRECT ??
1739 014334 001404          BEQ      1$
1740 014336 010037 001170          MOV      R0,$TMP4      ; NO
1741 014342 104102          FRROR    102
1742 014344 000415          BR       3$
1743
1744 014346 004537 004312          1$: JSR      R5,CHECK4      ; DATA CORRECT ??
1745 014352 014574 014604          PDATIO,PDAT00
1746 014356 001410          BEQ      3$            ; BR IF SO.
1747 014360 004537 004312          JSR      R5,CHECK4      ; NO, DID IT DO MODE 0.
1748 014364 014564 014604          PPAT10,PDAT00
1749 014370 001002          BNE      2$            ; NO.
1750 014372 104103          ERROR    103          ; YES, EXECUTED MODE 0.
1751 014374 000401          SKP1
1752 014376 104104          2$: ERROR    104          ; DATA INCORRECT.
1753 014400 000505          3$: BR       PDONE
1754
1755          ;TRAP TO HERE ON A BUS-ERROR.
1756
1757 014402 021627 014310          PERR0: CMP      (SP),#P3      ; DID IT INDEX AT ALL ??
1758 014406 001405          BEQ      10$           ; NO, CHECK IT OUT.
1759 014410 021627 014312          CMP      (SP),#P4      ; YES, DID IT DO MODE 7?
1760 014414 001445          BEQ      7$            ; YES.
1761 014416 000137 004224          JMP      @TRP04        ; ???
1762
1763 014422 020027 014333          10$: CMP      R0,#PDATIO-241 ;WAS IT FSRC MODE 1
1764 014426 001414          BEQ      1$
1765 014430 020027 014343          CMP      R0,#PDATIO-241+10 ;WAS IT FSRC MODE 2
1766 014434 001415          BEQ      2$

```

1767	014436	020027	014335			CMP	R0,#PDATIO-241+2		;WAS IT FSRC MODE 3
1768	014442	001416				BEQ	3\$		
1769	014444	020027	014323			CMP	R0,#PDATIO-241-10		;WAS IT FSRC MODE 4
1770	014450	001417				BEQ	4\$		
1771	014452	020027	014331			CMP	R0,#PDATIO-241-2		;WAS IT FSRC MODE 5
1772	014456	001420				BEQ	5\$		
1773									
1774	014460	012737	000321	001174	1\$:	MOV	#321,@#STMP6		
1775	014466	000403				SKP3			
1776	014470	012737	000322	001174	2\$:	MOV	#322,@#STMP6		
1777	014476	000403				SKP3			
1778	014500	012737	000323	001174	3\$:	MOV	#323,@#STMP6		
1779	014506	000403				SKP3			
1780	014510	012737	000324	001174	4\$:	MOV	#324,@#STMP6		
1781	014516	000403				SKP3			
1782	014520	012737	000325	001174	5\$:	MOV	#325,@#STMP6		
1783	014526	000403				SKP3			
1784	014530	012737	000327	001174	7\$:	MOV	#327,#STMP6		
1785	014536	012737	000627	001176		MOV	#627,@#STMP7		;REPORT FSRC
1786	014544	012737	000326	001200		MOV	#326,@#STMP10		;FLOWS FAILURE.
1787	014552	011637	001164			MOV	(SP),@#STMP2		
1788	014556	022626				CMP	(SP)+,(SP)+		
1789	014560	104101				ERROR	101		
1790	014562	000414				BR	PDONE		
1791									
1792	014564	177777	177777	177777	PPATIO:	.WORD	-1,-1,-1,-1		
1793									
1794	014574	010421	031463	052525	PDATIO:	.WORD	010421,031463,052525,073567		
1795									
1796	014604	000000	000000	000000	PDATIO:	.WORD	0,0,0,0		
1797									
1798	014614				PDONE:				
(1)	014614	104412				CLRFPS			:: CLEAR FP STATUS...
(3)	014616	000400				BR	TS123		::...AND PROCEED.

```
1805 .....  
(3) *TEST 23 LDD -- FSRC MODE 7 (INDEX DEFERRED)  
(4) *  
(4) * THIS IS A TEST OF FSRC MODE 7, INDEX DEFERRED.  
(4) *  
(3) .....  
(2) 014620 000004 TST23: SCOPE  
1806 014622 Q1: LUPERR ;: LOOP HERE ON ERROR !F SWR9 = 1.  
(1) 014622 104411 MOV #QERRO,ERRVEC ; SET BUS TRAP.  
1807 014624 012737 015022 000004 SETD  
1808 014632 170011 MOV #Q2,$TMP2  
1809 014634 012737 014674 001164 MOV #QDAT10,$TMP4  
1810 014642 012737 015206 001170 MOV #QDAT00,$TMP5  
1811 014650 012737 015176 001172 MOV #QPAT10,R0  
1812 014656 012700 015156 LDD (R0),AC0 ;: LOAD AC0  
(1) 014662 172410 MOV #QPAT20-241,R0  
1813 014664 012700 014725 MOV R0,$TMP3  
1814 014670 010037 001166 Q2: LDD @241(R0),AC0 ; TEST IT.  
1815 014674 172470 000241 Q3=Q2+2  
1816 014676 Q4: CLC  
1817 014700 000241 CLC  
1818 014702 000241 MOV R0,-(SP) ; SAVE R0 AFTER.  
1819 014704 010046 MOV #QDAT00,R0  
1820 014706 012700 015176 STD AC0,(R0) ;: STORE AC0  
(1) 014712 174010 MOV (SP)+,R0 AND RECALL R0.  
1821 014714 012600  
1822  
1823 014716 020027 014725 CMP R0,#QPAT20-241 ; R0 CORRECT ??  
1824 014722 001404 BEQ 1$ ; BR IF SO.  
1825 014724 010037 001170 MOV R0,$TMP4  
1826 014730 104106 ERROR 106 ; R0 INCORRECT.  
1827 014732 000431 BR 4$  
1828  
1829 014734 004537 004312 1$: JSR R5,CHECK4 ; DATA RIGHT ??  
1830 014740 015206 015176 QDAT10,QDAT00  
1831 014744 001425 BEQ 5$ ; BR IF SO.  
1832 014746 004537 004312 JSR R5,CHECK4 ; DID IT DO MODE 0 ??  
1833 014752 015156 015176 QPAT10,QDAT00  
1834 014756 001406 BEQ 2$ ; YES.  
1835 014760 004537 004312 JSR R5,CHECK4 ; DID IT DO MODE 6 ??  
1836 014764 015166 015176 QPAT20,QDAT00  
1837 014770 001405 BEQ 3$ ; YES.  
1838 014772 000411 BR 4$ ; BR IF NEITHER.  
1839  
1840 014774 012737 000320 001174 2$: MOV #320,$TMP6  
1841 015002 000403 SKP3  
1842 015004 012737 000326 001174 3$: MOV #326,$TMP6  
1843 015012 104107 ERROR 107 ; EXECUTED MODE 0 OR 6.  
1844 015014 000401 SKP1  
1845 015016 104110 4$: ERROR 110 ; DATA INCORRECT.  
1846 015020 000476 5$: BR QDONE  
1847  
1848 ; TRAP TO HERE ON A BUS-ERROR FAILURE  
1849  
1850 015022 021627 014676 QERRO: CMP (SP),#Q3  
1851 015026 001402 BEQ 1$
```

```

1852 015030 000137 004224      JMP      @TRP04
1853
1854 015034 020027 014725      1$:    CMP      R0,#QPAT20-241 ;WAS IT FSRC MODE 1 ??
1855 015040 001003                BNE
1856 015042 012737 000321 001174      MOV      #321,@$TMP6
1857 015050 020027 014735      2$:    CMP      R0,#QPAT20-241+10 ;WAS IT FSRC 2
1858 015054 001003                BNE
1859 015056 012737 000322 001174      MOV      #322,@$TMP6
1860 015064 020027 014727      3$:    CMP      R0,#QPAT20-241+2 ;WAS IT FSRC 3
1861 015070 001003                BNE
1862 015072 012737 000323 001174      MOV      #323,@$TMP6
1863 015100 020027 014715      4$:    CMP      R0,#QPAT20-241-10 ;WAS IT FSRC 4
1864 015104 001003                BNE
1865 015106 012737 000324 001174      MOV      #324,@$TMP6
1866 015114 020027 014723      5$:    CMP      R0,#QPAT20-241-2 ;WAS IT FSRC 5
1867 015120 001003                BNE
1868 015122 012737 000325 001174      MOV      #325,@$TMP6
1869
1870 015130 012737 000627 001176      6$:    MOV      #627,@$TMP7 ;REPORT FSRC FAILURE
1871 015136 012737 000327 001200      MOV      #327,@$TMP10
1872 015144 011637 001164      MOV      (SP),@$TMP2
1873 015150 022626                CMP      (SP)+,(SP)+
1874 015152 104105                ERROR   105
1875 015154 000420                BR      QDONE
1876
1877 015156 177777 177777 177777  QPAT10: .WORD  -1,-1,-1,-1
1878
1879 015166 015206 052525 052525  QPAT20: .WORD  QDAT10,052525,052525,052525
1880
1881 015176 000000 000000 000000  QDAT00: .WORD  0,0,0,0
1882
1883 015206 073567 052525 031463  QDAT10: .WORD  073567,052525,031463,010421
1884
1885 015216                QDONE:
(1) 015216 104412                CLRFP5
(3) 015220 000400                BR      TST24 ;: CLEAR FP STATUS...
;:...AND PROCEED.

```


1903
 (3)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2)

1904	015222	000004		
1905	015224	012737	015532	000244
1906	015232	012727	177777	
1907	015236	177777		
1908	015240	012700	177777	
1909	015244	000401		
1910	015246	005000		
1911	015250	010037	015550	
1912	015254	010037	015552	
1913	015260	010037	015554	
1914	015264	010037	015556	
1915	015270	012737	015440	001164
1916	015276	005003		
1917	015300	012700	015560	
1918	015304	012704	000204	
1919	015310	004737	015416	
1920	015314	012700	015570	
1921	015320	012704	000200	
1922	015324	004737	015416	
1923	015330	012700	015600	
1924	015334	012700	000210	
1925	015340	004737	015416	
1926	015344	012700	015610	
1927	015350	012704	000214	
1928	015354	004737	015416	
1929	015360	005237	015236	
1930	015364	001730		
1931				
1932	015366	172427	000000	
1933	015372	012701	004200	
1934	015376	170101		
1935	015400	012704	104214	
1936	015404	012703	000014	
1937	015410	004737	015424	
1938	015414	000501		
1939				
1940	015416	012701	000200	
1941	015422	170101		

```

*****
*TEST 24      TEST FZ, FN, AND FIUV BITS USING LDD
*
* THIS IS A TEST OF THE (BUT EZBT Y8) FORK, THE
* (BUT ENBT) FORK AND (BUT FIUV) FORK IN THE
* LOAD INSTRUCTION FLOWS.
* EACH OF THE PATTERNS:
*      0
*      +NUM
*      -NUM
*      -0
* IS LOADED TWICE, ONCE WITH AC<>0 THEN
* WITH AC=0. AFTER EACH LOAD THE FPS IS
* CHECK TO INSURE THAT CONTROL WAS PASSED
* THROUGH WITH THE FORKS PROPERLY.
*****
TST24:  SCOPE
        MOV      #U244,FPVEC      ; FOR THE FIUV PART.
        MOV      #-1,(PC)+
U1:     -1                          ; LOOP FLAG.
        MOV      #-1,R0           ; BACKGROUND = 1'S FIRST...
        SKP1
U2:     CLR      R0                ;...THEN 0'S.
        MOV      R0,UPAT0
        MOV      R0,UPAT0+2
        MOV      R0,UPAT0+4
        MOV      R0,UPAT0+6
        MOV      #U5,$TMP2
        CLR      R3                ; USE AS TRAP FLAG.
        MOV      #UPAT1,R0         ; DATA = 0'S
        MOV      #204,R4           ; EXP D AND Z
        JSR      PC,U3
        MOV      #UPAT2,R0         ; DATA = +N
        MOV      #200,R4           ; EXP D ONLY.
        JSR      PC,U3
        MOV      #UPAT3,R0         ; DATA = -N
        MOV      #210,R0           ; EXP D AND N.
        JSR      PC,U3
        MOV      #UPAT4,R0         ; DATA = -0
        MOV      #214,R4           ; EXP D, N, AND Z (FIUV - 0).
        JSR      PC,U3
        INC      U1
        BEQ      U2                ; GO 'ROUND AGAIN...
        LDD      #0,AC0             ; NOW, CLEAR AC0...
        MOV      #4200,R1          ;...AND SEE IF FIUV WILL...
        LDFPS   R1                 ;...TRAP ON -0.
        MOV      #104214,R4        ; EXP FPS ???
        MOV      #14,R3            ; EXP FEC ???
        JSR      PC,U4
        BR       UDONE             ;...THEN EXIT.
U3:     MOV      #200,R1
        LDFPS   R1                 ; SET D ONLY.
  
```

```

1942 015424          U4:      LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
(1) 015424 104411      MOV      #UPAT0,R1
1943 015426 012701 015550      LDD      (R1),AC0          ; SET BACKGROUND.
1944 015432 172411      240,240
1945 015434 000240 000240      U5:      LDD      (R0),AC0          ; CHANGE TO DATA PATTERN.
1946 015440 172410      240,240
1947 015442 000240 000240      TST      R3              ; EXPECTING A UV TRAP ??
1948 015446 005703      BNE      UERR2          ; YES, BUT IT DIDN'T HAPPEN.
1949 015450 001014      STFPS    R5              ; NO, FPS => R5.
1950 015452 170205      CMP      R4,R5
1951 015454 020405      BNE      UERR1          ; FPS IS WRONG.
1952 015456 001001      RTS      PC
1953 015460 000207
1954
1955 015462 010037 001172      UERR1:   MOV      R0,$TMP5
1956 015466 010437 001170      MOV      R4,$TMP4
1957 015472 010537 001166      MOV      R5,$TMP3
1958 015476 104125      ERROR    125            ; FPS INCORRECT ETC...
1959 015500 000447      BR       UDONE
1960 015502 104126      UERR2:   ERROR    126            ; -0 DIDN'T TRAP WITH FIUV = 1.
1961 015504 000445      BR       UDONE
1962 015506 104127      UERR3:   ERROR    127            ; -0 TRAPPED WITH FIUV = 0.
1963 015510 000443      BR       UDONE
1964 015512 010037 001172      UERR4:   MOV      R0,$TMP5
1965 015516 010337 001170      MOV      R3,$TMP4
1966 015522 010237 001166      MOV      R2,$TMP3
1967 015526 104130      ERROR    130            ; FEC INCORRECT AFTER TRAP.
1968 015530 000433      BR       UDONE
1969
1970 015532 022626      U244:   CMP      (SP)+,(SP)+
1971 015534 005703      TST      R3              ; TRAP EXPECTED ??
1972 015536 001763      BEQ      UERR3          ; NO, ERROR.
1973 015540 170302      2$:     STST     R2              ; YES, GET FEC => R2.
1974 015542 020302      CMP      R3,R2
1975 015544 001362      BNE      UERR4          ; BR IF IT'S WRONG.
1976 015546 000207      3$:     RTS      PC
1977
1978 015550 000000 000000 000000  UPAT0:  0,0,0,0
1979
1980 015560 000000 000000 000000  UPAT1:  0,0,0,0          ; PATTERN 1 = U
1981
1982 015570 010421 114631 125252  UPAT2:  010421,114631,125252,177777      ; POS NUM.
1983
1984 015600 114631 135673 146314  UPAT3:  114631,135673,146314,167356      ; NEG NUM.
1985
1986 015610 100000 000000 000000  UPAT4:  100000,0,0,0          ; NEG ZERO.
1987
1988 015620          UDONE:
(1) 015620 104412      CLRFPs          ;; CLEAR FP STATUS...
(3) 015622 000400      BR       TST25      ;...AND PROCEED.

```

```

1996
(3)
(4)
(4)
(4)
(4)
(3)
(2) 015624 000004
1997 015626 012737 016156 001172
1998 015634 012737 016156 001176
1999 015642 012737 016166 001174
2000 015650 012737 016176 001200
2001 015656 170011
2002 015660 104411
2003 015662 012700 016156
(1) 015666 172410
2004 015670 012700 016166
2005 015674 172010
2006 015676 012737 015674 001164
2007 015704 004737 016026
2008
2009 015710 104411
2010 015712 012700 016156
(1) 015716 172410
2011 015720 012700 016166
2012 015724 170001
2013 015726 172010
2014 015730 012737 015726 001164
2015 015736 004737 016050
2016
2017 015742 104411
2018 015744 012700 016156
(1) 015750 172410
2019 015752 012700 016166
2020 015756 173010
2021 015760 012737 015756 001164
2022 015766 004737 016026
2023
2024 015772 104411
2025 015774 012700 016156
(1) 016000 172410
2026 016002 012700 016166
2027 016006 170001
2028 016010 173010
2029 016012 012737 016010 001164
2030 016020 004737 016050
2031
2032 016024 000470
2033
2034 016026
2035 016026 112737 000104 053211
2036 016034 112737 000104 053272
2037 016042 012704 000204
2038 016046 000410
2039 016050
2040 016050 112737 000106 053211

```

```

*****
*TEST 25      ADD AND SUB -- FSRC MODE 1, WITH BOTH OPERANDS = 0
*
* THIS IS A TEST OF ADD AND SUB WITH BOTH OPERANDS = 0.
*   0 + 0 = 0 (HOPEFULLY)
*
*****
TST25:  SCOPE
W1:     MOV     #WOP1,$TMP5      ; SET ERROR POINTERS.
        MOV     #WOP1,$TMP7
        MOV     #WOP2,$TMP6
        MOV     #WRES,$TMP10    ; ACTUAL RESULT.
        SETD                    ; SET DOUBLE.
        LUPERR                   ;; LOOP HERE ON ERROR IF SWR9 = 1.
        MOV     #WOP1,R0
        LDD     (R0),ACO         ;; LOAD ACO
        MOV     #WOP2,R0
1$:     ADDD    (R0),ACO         ; ADDD
        MOV     #1$, $TMP2
        JSR    PC,WERR1        ; CHECK FOR ERRORS.
        LUPERR                   ;; LOOP HERE ON ERROR IF SWR9 = 1.
        MOV     #WOP1,R0
        LDD     (R0),ACO         ;; LOAD ACO
        MOV     #WOP2,R0
2$:     SETF                    ; ADD FLOAT
        ADDF    (R0),ACO
        MOV     #2$, $TMP2
        JSR    PC,WERR2
        LUPERR                   ;; LOOP HERE ON ERROR IF SWR9 = 1.
        MOV     #WOP1,R0
        LDD     (R0),ACO         ;; LOAD ACO
        MOV     #WOP2,R0
3$:     SUBD    (R0),ACO         ; SUB DOUBLE.
        MOV     #3$, $TMP2
        JSR    PC,WERR3
        LUPERR                   ;; LOOP HERE ON ERROR IF SWR9 = 1.
        MOV     #WOP1,R0
        LDD     (R0),ACO         ;; LOAD ACO
        MOV     #WOP2,R0
4$:     SETF                    ; SUB FLOATING.
        SUBF    (R0),ACO
        MOV     #4$, $TMP2
        JSR    PC,WERR4
        BR     WDONE
WERR1:
WERR3:  MOVB    #'D,EM140X      ; ERROR CHECK ADD...
        MOVB    #'D,EM141X      ; ...AND/OR SUB DOUBLE.
        MOV     #204,R4
        SKP3+3+2
WERR2:
WERR4:  MOVB    #'F,EM140X      ; ERROR CHECK ADD...

```

2041 016056 112737 000106 053272
2042 016064 012704 000004
2043
2044 016070 012637 016154
2045 016074 170205
2046 016076 170011
2047 016100 012700 016176
(1) 016104 174010
2048 016106 004537 004312
2049 016112 016156 016176
2050 016116 001002
2051 016120 020405
2052 016122 001413
2053 016124 010437 001166 2\$:
2054 016130 010537 001170
2055 016134 027727 163024 173010
2056 016142 001402
2057 016144 104140
2058 016146 000401
2059 016150 104141
2060 016152 000137 3\$:
2061 016154 000000 4\$:
2062 0
2063 016156 000000 000000 000000 WOP1: .FLT4 0
2064 016166 000000 000000 000000 WOP2: .FLT4 0
2065 016176 000000 000000 000000 WRES: .FLT4 0 ; RECEIVED RESULT.
2066
2067 016206 WDONE:
(1) 016206 104412 CLRFP
(3) 016210 000400 BR TST26

MOV #F,EM141X ;...AND/OR SUB SINGLE.
MOV #4,R4
MOV (SP)+,4\$; IN CASE LOOP IS SET.
STFPS R5 ; FPS => R5.
SETD ; RESTORE DOUBLE MODE.
MOV #WRES,R0
STD ACO,(R0) ;; STORE ACO
JSR R5,CHECK4 ; COMPARE...
WOP1,WRES ;...EXPECTED VS RECEIVED.
BNE 2\$; BR IF WRONG.
CMP R4,R5 ; CHECK FPS.
BEQ 3\$
MOV R4,\$TMP3
MOV R5,\$TMP4
CMP @ \$TMP2,#SUBD+10
BEQ :+6
ERROR 140 ; ADD ERROR
SKP1
ERROR 141 ; SUB ERROR
JMP @ (PC)+
0
WOP1: .FLT4 0
WOP2: .FLT4 0
WRES: .FLT4 0 ; RECEIVED RESULT.
WDONE:
CLRFP
BR TST26 ; CLEAR FP STATUS...
;...AND PROCEED.

```

2076 .....
(3) *TEST 26      ADDD AND SUBD -- FSRC MODE 1, WITH SRC OPERAND = 0
(4) *
(4) * THIS IS A TEST OF ADDD AND SUBD WITH JUST THE SRC = 0.
(4) *           N + 0 = N
(4) *           N - 0 = N
(3) .....
(2) 016212 000004 TST26: SCOPE
2077 016214 012737 016556 001172 X1:  MOV    #XOP1,$TMP5      ; SRC OPERAND.
2078 016222 012737 016606 001200     MOV    #XRES,$TMP10     ; ACTUAL RESULT.
2079 016230 112737 000104 053211     MOVB  #'D,EM140X
2080 016236 112737 000104 053272     MOVB  #'D,EM141X
2081 016244 170011     SE D      ; SET DOUBLE.
2082 016246 104411     LUPERR   ;; LOOP HERE ON ERROR IF SWR9 = 1.
2083 016250 012700 016566     MOV    #XOP2,R0
(1) 016254 172410     LDD    (R0),AC0        ;; LOAD ACO
2084 016256 010037 001174     MOV    R0,$TMP6
2085 016262 010037 001176     MOV    R0,$TMP7
2086 016266 012700 016556     MOV    #XOP1,R0
2087 016272 172010 3$:  ADDD   (R0),AC0        ; N + 0
2088 016274 012737 016272 001164     MOV    #1$, $TMP2
2089 016302 004737 016450     JSR    PC,XERR1        ; CHECK FOR ERRORS.
2090
2091 016306 104411     LUPERR   ;; LOOP HERE ON ERROR IF SWR9 = 1.
2092 016310 012700 016576     MOV    #XOP3,R0
(1) 016314 172410     LDD    (R0),AC0        ;; LOAD ACO
2093 016316 010037 001174     MOV    R0,$TMP6
2094 016322 010037 001176     MOV    R0,$TMP7
2095 016326 012700 016556     MOV    #XOP1,R0
2096 016332 172010 2$:  ADDD   (R0),AC0        ; -N + 0
2097 016334 012737 016332 001164     MOV    #2$, $TMP2
2098 016342 004737 016456     JSR    PC,XERR2
2099
2100 016346 104411     LUPERR   ;; LOOP HERE ON ERROR IF SWR9 = 1.
2101 016350 012700 016566     MOV    #XOP2,R0
(1) 016354 172410     LDD    (R0),AC0        ;; LOAD ACO
2102 016356 010037 001174     MOV    R0,$TMP6
2103 016362 010037 001176     MOV    R0,$TMP7
2104 016366 012700 016556     MOV    #XOP1,R0
2105 016372 173010 3$:  SUBD   (R0),AC0        ; N - 0
2106 016374 012737 016372 001164     MOV    #3$, $TMP2
2107 016402 004737 016450     JSR    PC,XERR3
2108
2109 016406 104411     LUPERR   ;; LOOP HERE ON ERROR IF SWR9 = 1.
2110 016410 012700 016576     MOV    #XOP3,R0
(1) 016414 172410     LDD    (R0),AC0        ;; LOAD ACO
2111 016416 010037 001174     MOV    R0,$TMP6
2112 016422 010037 001176     MOV    R0,$TMP7
2113 016426 012700 016556     MOV    #XOP1,R0
2114 016432 173010 4$:  SUBD   (R0),AC0        ; -N - 0
2115 016434 012737 016432 001164     MOV    #4$, $TMP2
2116 016442 004737 016456     JSR    PC,XERR4
2117
2118 016446 000463     BR     XDONE
2119

```

2120	016450				XERR1:				
2121	016450	012704	000200		XERR3:	MOV	#200,R4	:	EXPECT POS NON-ZERO.
2122	016454	000402				SKP2			
2123	016456				XERR2:				
2124	016456	012704	000210		XERR4:	MOV	#210,R4	:	EXPECT NEG NON-ZERO.
2125									
2126	016462	012637	016554			MOV	(SP)+,4\$:	IN CASE LOOP IS SET.
2127	016466	170205				STFPS	R5	:	FPS => R5.
2128	016470	170011				SETD		:	INSURE DOUBLE MODE.
2129	016472	012700	016606			MOV	#XRES,R0		
(1)	016476	174010				STD	ACO,(R0)	::	STORE ACO
2130	016500	013737	001176	016512		MOV	\$TMP7,1\$:	SET EXP POINTER.
2131	016506	004537	004312			JSR	R5,CHECK4		
2132	016512	000000	016606		1\$:	O,XRES		:	EXP VS RECD.
2133	016516	001002				BNE	2\$		
2134	016520	020405				CMP	R4,R5	:	CHECK FPS.
2135	016522	001413				BEQ	3\$		
2136	016524	010437	001166		2\$:	MOV	R4,\$TMP3		
2137	016530	010537	001170			MOV	R5,\$TMP4		
2138	016534	027727	162424	173010		CMP	@\$TMP2,#SUBD+10		
2139	016542	001402				BEQ	+6		
2140	016544	104140				ERROR	140	:	ADD ERROR
2141	016546	000401				SKP1			
2142	016550	104141				ERROR	141	:	SUB ERROR
2143	016552	000137			3\$:	JMP	@(PC)+		
2144	016554	000000			4\$:	O			
2145									
2146	016556	000000	000000	000000	XOP1:	.FLT4	0		
2147	016566	010421	021042	031463	XOP2:	010421,021042,031463,042104		:	POS N.
2148	016576	104210	114631	125252	XOP3:	104210,114631,125252,135673		:	NEG NUM.
2149	016606	000000	000000	000000	XRES:	.FLT4	0	:	RECEIVED RESULT.
2150									
2151	016616				XDONE:				
(1)	016616	104412				CLRFPS		::	CLEAR FP STATUS...
(3)	016620	000400				BR	TST27	::AND PROCEED.

```
2161 .....  
(3) *TEST 27 SUBD -- FSRC MODE 1, WITH DST OPERAND 0  
(4) *  
(4) * THIS IS A TEST OF THE SUBD WITH THE DST OPERAND - 0.  
(4) * BOTH POSITIVE AND NEGATIVE SRC OPERANDS ARE TRIED.  
(4) * 0 - (+N) = -N  
(4) * 0 - (-N) = +N  
(4) *  
(3) .....  
(2) 016622 000004 TST27: SCOPE  
2162 016624 012727 177777 MOV # -1, (PC)+  
2163 016630 177777 Y1: -1 ; LOOP FLAG.  
2164 016632 012737 000210 001170 MOV #210, STMP4 ; EXPECT NEG NON-ZERO STATUS.  
2165 016640 012737 017032 001172 MOV #YOP1, STMP5 ; POS NUM (SRC).  
2166 016646 012737 017042 001176 MOV #YOP2, STMP7 ; NEG NUM (EXP RESULT).  
2167 016654 000411 BR Y3  
2168 016656 012737 000200 001170 Y2: MOV #200, STMP4 ; EXPECT POS NON-ZERO STATUS.  
2169 016664 012737 017042 001172 MOV #YOP2, STMP5 ; REVERSE THE VALUES.  
2170 016672 012737 017032 001176 MOV #YOP1, STMP7  
2171  
2172 016700 012737 017022 001174 Y3: MOV #YOP0, STMP6 ; DST OPERAND.  
2173 016706 012737 017052 001200 MOV #YRES, STMP10 ; RECEIVED RESULT.  
2174 016714 012737 000104 053272 MOVB #D, EM141X  
2175 016722 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
2176 016724 170011 SETD ;: SET DOUBLE MODE.  
2177 016726 172427 000000 LDD #0, ACO ;: CLEAR ACO.  
2178 016732 013700 001172 MOV STMP5, R0  
2179 016736 173010 Y4: SUBD (R0), ACO ;: 0 - (+N) THEN 0 - (-N).  
2180 016740 170205 STFPS R5 ;: STATUS => R5.  
2181 016742 012700 017052 MOV #YRES, R0  
(1) 016746 174010 STD ACO, (R0) ;: STORE ACO  
2182 016750 013737 001176 016762 MOV STMP7, 1$ ;: SET EXP POINTER.  
2183 016756 004537 004312 JSR R5, CHECK4 ;: COMPARE  
2184 016762 000000 017052 1$: 0, YRES ;: EXP VS RECD.  
2185 016766 001003 RNE 2$  
2186 016770 023705 001170 #P STMP4, R5 ;: CHECK FPS TOO.  
2187 016774 001406 #Q 3$  
2188 016776 012737 016736 001164 2$: MOV #Y4, STMP2  
2189 017004 010537 001166 MOV R5, STMP3  
2190 017010 104141 ERROR 141 ;: SUBD ERROR.  
2191 017012 005237 016630 3$: INC Y1  
2192 017016 001717 BEQ Y2 ;: REVERSE AND DO AGAIN...  
2193 017020 000420 BR YDONE ;: ...AND QUIT.  
2194  
2195 017022 000000 000000 000000 YOP0: .FLT4 0  
2196 017032 063146 052525 042104 YOP1: 063146, 052525, 042104, 167356 ;: POS NUM...  
2197 017042 163146 052525 042104 YOP2: 163146, 052525, 042104, 167356 ;: ...AND IT'S NEG.  
2198 017052 000000 000000 000000 YRES: .FLT4 0 ;: RECEIVED RESULT.  
2199  
2200 017062 YDONE:  
(1) 017062 104412 CLRFPs ;: CLEAR FP STATUS...  
(3) 017064 000400 BR TST30 ;: ...AND PROCEED.
```

```

2210 *****
(3) *TEST 30 ADDD -- FSRC MODE 1, WITH DST OPERAND = 0
(4) *
(4) * THIS IS A TEST OF THE ADDD WITH THE DST OPERAND = 0.
(4) * BOTH POSITIVE AND NEGATIVE SRC OPERANDS ARE TRIED.
(4) * 0 + (+N) = +N
(4) * 0 + (-N) = -N
(3) *****
(2) 017066 000004 TST30: SCOPE
2211 017070 012727 177777 MOV #-1,(PC)+
2212 017074 177777 Z1: -1 ; LOOP FLAG.
2213 017076 012737 000200 001170 MOV #200,$TMP4 ; EXPECT POS NON-ZERO STATUS.
2214 017104 012737 017270 001172 MOV #ZOP1,$TMP5 ; POS NUM (SRC).
2215 017112 000406 BR Z3
2216 017114 012737 000210 001170 Z2: MOV #210,$TMP4 ; EXPECT NEG NON-ZERO STATUS.
2217 017122 012737 017300 001172 MOV #ZOP2,$TMP5 ; REVERSE THE VALUES.
2218
2219 017130 012737 017260 001174 Z3: MOV #ZOP0,$TMP6 ; DST OPERAND.
2220 017136 013737 001172 001176 MOV $TMP5,$TMP7 ; EXP RESULT.
2221 017144 012737 017310 001200 MOV #ZRES,$TMP10 ; RECEIVED RESULT.
2222 017152 112737 000104 053211 MOVB #D,EM140X
2223 017160 104411 ;: LOOP HERE ON ERROR IF SWR9 1.
2224 017162 170011 SETD ; SET DOUBLE MODE.
2225 017164 172427 000000 LDD #0,ACO ; CLEAR ACO.
2226 017170 013700 001172 MOV $TMP5,R0
2227 017174 172010 Z4: ADDD (R0),ACO ; 0 + (+N) THEN 0 + (-N).
2228 017176 170205 STFPS R5 ; STATUS => R5.
2229 017200 012700 017310 MOV #ZRES,R0
(1) 017204 174010 STD ACO,(R0) ;: STORE ACO
2230 017206 013737 001176 017220 MOV $TMP7,1$
2231 017214 004537 004312 JSR R5,CHECK4 ; COMPARE.
2232 017220 000000 017310 1$: O,ZRES ; EXP VS RECD.
2233 017224 001003 BNE 2$
2234 017226 023705 001170 CMP $TMP4,R5 ; CHECK FPS TOO.
2235 017232 001406 BEQ 3$
2236 017234 012737 017174 001164 2$: MOV #Z4,$TMP2
2237 017242 010537 001166 MOV R5,$TMP3
2238 017246 104140 ERROR 140 ; ADDD ERROR.
2239 017250 005237 017074 3$: INC Z1
2240 017254 001717 BEQ Z2 ; REVERSE AND DO AGAIN...
2241 017256 000420 BR ZDONE ;...AND QUIT.
2242
2243 017260 000000 000000 000000 ZOP0: .FLT4 0
2244 017270 031463 010421 146314 ZOP1: 031463,010421,146314,156735 ; POS N.
2245 017300 156735 167356 135673 ZOP2: 156735,167356,135673,146314 ; NEG N.
2246 017310 000000 000000 000000 ZRES: .FLT4 0 ; RECEIVED RESULT.
2247
2248 017320 ZDONE:
(1) 017320 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 017322 000400 BR TST31 ;:...AND PROCEED.
  
```



```
2257 (3) *****
(4) *TEST 31 ADD AND ADDF (EPOXNENTS EQUAL), AND ROUND\TRUNK (BUT FT)
(4) *
(4) * THIS IS A TEST OF THE ADD INSTRUCTIONS WITH THE OPERANDS
(4) * HAVING EQUAL EXPONENTS (ALIGNMENT NOT REQUIRED).
(4) * THE ROUND/TRUNCATE FLOW (BUT FT) IS ALSO TESTED HERE.
(3) *****
(2) 017324 000004 TST31: SCOPE
2258 017326 AA1:
(1) 017326 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
2259 017330 012700 000240 MOV #240,R0
2260 017334 170100 LDFPS R0 ; SET FD AND FT.
2261 017336 012737 017766 001172 MOV #AAOP0,$TMP5 ; DST OPERAND FOR ERRORS.
2262 017344 012737 017756 001176 MOV #AASUM,$TMP7 ; DITTO RECEIVED SUM.
2263 017352 012737 017400 001164 MOV #AA2,@#$TMP2
2264 017360 012700 017766 MOV #AAOP0,R0
(1) 017364 172410 LDD (R0),ACO ;; LOAD ACO
2265 017366 012700 017776 MOV #AAOP1,R0
2266 017372 010037 001170 MOV R0,$TMP4 ; SRC OPERAND.
2267 017376 000240 NOP
2268 017400 172010 AA2: ADDD (R0),ACO ; ADD OP0 + OP1 AND TRUNCATE.
2269 017402 170205 STFPS R5
2270 017404 010537 001166 MOV R5,$TMP3
2271 017410 170011 SETD
2272 017412 012700 017756 MOV #AASUM,R0
(1) 017416 174010 STD ACO,(R0) ;; STORE ACO
2273 017420 012737 020006 001174 MOV #AAOP2,$TMP6 ; EXP POINTER.
2274 017426 004537 004312 JSR R5,CHECK4
2275 017432 020006 017756 AAOP2,AASUM ; EXP VS RECD.
2276 017436 001410 BEQ AA10 ; BR IF OK.
2277 017440 004537 004312 JSR R5,CHECK4 ; IT'S NOT...
2278 017444 020016 017756 AAOP3,AASUM ; ...DID TRUNCATE FAIL ??
2279 017450 001402 BEQ 1$ ; YES.
2280 017452 104145 ERROR 145 ; NO, SUM INCORRECT.
2281 017454 000401 SKP1
2282 017456 104146 1$: ERROR 146 ; DIDN'T TRUNCATE.
2283
2284 ;NOW TEST DOUBLE FLOATING ROUND MODE.
2285
2286 017460 AA10:
(1) 017460 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
2287 017462 012700 000200 MOV #200,R0
2288 017466 170100 LDFPS R0 ; CLEAR FT BIT.
2289 017470 012737 017520 001164 MOV #AA11,@#$TMP2 ; FOR ERROR 147 TEXT.
2290 017476 112737 000104 053423 MOVB #'D,EM147+3
2291 017504 012700 017766 MOV #AAOP0,R0
(1) 017510 172410 LDD (R0),ACO ;; LOAD ACO
2292 017512 012700 017776 MOV #AAOP1,R0
2293 017516 000240 NOP
2294 017520 172010 AA11: ADDD (R0),ACO ; ADD OP0 + OP1 AND ROUND.
2295 017522 170205 STFPS R5
2296 017524 010537 001166 MOV R5,$TMP3
2297 017530 170011 SETD
2298 017532 012700 017756 MOV #AASUM,R0
(1) 017536 174010 STD ACO,(R0) ;; STORE ACO
```

2299	017540	012737	020016	001174	MOV	#AAOP3,\$TMP6	
2300	017546	004537	004312		JSR	R5,CHECK4	
2301	017552	020016	017756		AAOP3,AASUM		; EXP VS RECD.
2302	017556	001417			BEQ	AA20	; BR IF OK.
2303	017560	004537	004312		JSR	R5,CHECK4	; IT'S NOT...
2304	017564	020006	017756		AAOP2,AASUM		...DID IT TRUNCATE ??
2305	017570	001407			BEQ	1\$; YES.
2306	017572	004537	004312		JSR	R5,CHECK4	; NO, DID IT ROUND F MODE ??
2307	017576	020026	017756		AAOP4,AASUM		
2308	017602	001404			BEQ	2\$; YES.
2309	017604	104145			ERROR	145	; SUM INCORRECT.
2310	017606	000401			SKP1		
2311	017610	104147		1\$:	ERROR	147	; DIDN'T ROUND.
2312	017612	000401			SKP1		
2313	017614	104150		2\$:	ERROR	150	; ROUNDED IN F MODE.
2314							
2315							
2316							
2317	017616						
(1)	017616	104411			AA20:		
2318	017620	012700	000200		LUPERR		:: LOOP HERE ON ERROR IF SWR9 1.
2319	017624	170100			MOV	#200,R0	
2320	017626	012737	017662	001164	LDFPS	R0	; FT = 0
2321	017634	112737	000106	053423	MOV	#AA21,@#\$TMP2	
2322	017642	012700	017766		MOVB	#'F,EM147+3	; FOR ERROR 147 TEXT.
(1)	017646	172410			MOV	#AAOP0,R0	
2323	017650	012700	020036		LDD	(R0),ACO	:: LOAD ACO
2324	017654	010037	001170		MOV	#AAOP5,R0	
2325	017660	170001			MOV	R0,\$TMP4	; SRC OPERAND.
2326	017662	172010			SETF		
2327	017664	170205		AA21:	ADDF	(R0),ACO	;ADDF OP0 + OP1 AND ROUND.
2328	017666	170011			STFPS	R5	
2329	017670	012700	017756		SETD		; RET TO DOUBLE
(1)	017674	174010			MOV	#AASUM,R0	
2330	017676	012737	020046	001174	STD	ACO,(R0)	:: STORE ACO
2331	017704	004537	004312		MOV	#AAOP6,\$TMP6	
2332	017710	020046	017756		JSR	R5,CHECK4	
2333	017714	001460			AAOP6,AASUM		; EXP VE RECD.
2334	017716	004537	004312		BEQ	AADONE	; BR IF OK.
2335	017722	020006	017756		JSR	R5,CHECK4	; IT'S NOT...
2336	017726	001407			AAOP2,AASUM		...DID IT TRUNCATE ??
2337	017730	004537	004312		BEQ	1\$; YES.
2338	017734	020016	017756		JSR	R5,CHECK4	; NO...
2339	017740	001404			AAOP3,AASUM		...DID IT ROUND DOUBLE ??
2340	017742	104145			BEQ	2\$; YES.
2341	017744	000401			ERROR	145	; SUM INCORRECT.
2342	017746	104147		1\$:	SKP1		
2343	017750	000401			ERROR	147	; DIDN'T ROUND.
2344	017752	104151		2\$:	SKP1		
2345	017754	000440			ERROR	151	; ROUNDED IN D MODE.
2346					BR	AADONE	
2347	017756	000000	000000	000000	AASUM:	.FLT4 0	; RECEIVED SUM.
2348	017766	000200	000000	000000	AAOP0:	000200,0,0,0	; EXP 1, MANTISSA 0
2349	017776	000200	000000	000000	AAOP1:	000200,0,0,1	; EXP 1, MANTISSA 1.
2350	020006	000400	000000	000000	AAOP2:	000400,0,0,0	; DOUBLE SUM (0+1), TRUNCATED.
2351	020016	000400	000000	000000	AAOP3:	000400,0,0,1	; DOUBLE SUM (0+1), ROUNDED.

2352 020026 000400 000000 100000 AAOP4: 000400,0,100000,0 ; DOUBLE SUM (0+1), ROUNDED F.
2353
2354 020036 000200 000001 000000 AAOP5: 000200,1,0,0 ; EXP 1, MANTISSA 1, SINGLE.
2355 020046 000400 000001 000000 AAOP6: 000400,1,0,0 ; SINGLE SUM (0+5), ROUNDED.
2356
2357 020056 AADONE:
(1) 020056 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 020060 000400 BR TST32 ;: ...AND PROCEED.

```

2369
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 020062 000004
2370
2371
2372
2373
2374 020064
(1) 020064 104411
2375 020066 112737 000104 053616
2376 020074 170011
2377 020076 012737 020126 001164
2378 020104 012700 020744
(1) 020110 172410
2379 020112 010037 001172
2380 020116 012700 020764
2381 020122 010037 001170
2382 020126 172010
2383 020130 170205
2384 020132 010537 001166
2385 020136 012700 020734
(1) 020142 174010
2386 020144 004537 004312
2387 020150 020764 020734
2388 020154 001407
2389 020156 012737 020764 001174
2390 020164 012737 020734 001176
2391 020172 104152
2392
2393
2394
2395
2396 020174
(1) 020174 104411
2397 020176 170011
2398 020200 012737 020230 001164
2399 020206 012700 020744
(1) 020212 172410
2400 020214 010037 001172
2401 020220 012700 020754
2402 020224 010037 001170
2403 020230 172010
2404 020232 170205
2405 020234 010537 001166
2406 020240 012700 020734
(1) 020244 174010
2407 020246 004537 004312
  
```

```

*****
*TEST 32      ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC)
*
*THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS
*AND THE ALIGN FDST EXPONENT ALGORITHM.
*THE ALIGNMENT CONSTANTS 25 FOR FLOATING, AND 57 FOR DOUBLE
*ARE VERIFIED.
*NOTE THAT E(AC) IS LESS THAN E(SRC) SO THE OPERAND IN AC
*(FDST) IS THE ONE TO BE ALIGNED.
*
*****
TST32: SCOPE
:
: FIRST EXPONENT DIFFERENCE = 58 (72 OCTAL).
: ALIGNMENT NOT POSSIBLE.
:
CC1:
LUPERR          ;; LOOP HERE ON ERROR IF SWR9 - 1.
MOVB #'D,EM152X
SETD            ; SET DOUBLE.
MOV #CC2,@$STMP2
MOV #CCP0,R0
LDD (R0),AC0    ;; LOAD AC0
MOV R0,$STMP5   ; DST OPERAND.
MOV #CCP2,R0
MOV R0,$STMP4   ; SRC OPERAND.
CC2: ADDD (R0),AC0 ; TEST INSTRUCTION
STFPS R5       ; GET FPS
MOV R5,$STMP3
MOV #CCSUM,R0
STD AC0,(R0)   ;; STORE AC0
JSR R5,CHECK4
CCP2,CCSUM    ; EXP VS RECD.
BEQ CC7      ; BR IF OK.
MOV #CCP2,$STMP6 ; EXPECTED.
MOV #CCSUM,$STMP7
ERROR 152    ; INCORRECT.
:
: NOW EXPONENT DIFF = 57 (71 OCT).
: MAXIMUM DOUBLE ALIGNMENT.
:
CC7:
LUPERR          ;; LOOP HERE ON ERROR IF SWR9 1.
SETD            ; SET DOUBLE.
MOV #CC8,@$STMP2
MOV #CCP0,R0
LDD (R0),AC0    ;; LOAD AC0
MOV R0,$STMP5   ; FSRC
MOV #CCP1,R0
MOV R0,$STMP4
CC8: ADDD (R0),AC0 ; TEST INSTRUCTION
STFPS R5       ; GET FPS
MOV R5,$STMP3
MOV #CCSUM,R0
STD AC0,(R0)   ;; STORE AC0
JSR R5,CHECK4
  
```

2408 020252 021034 020734
2409 020256 001407
2410 020260 012737 021034 001174
2411 020266 012737 020734 001176
2412 020274 104152
2413
2414
2415
2416
2417 020276
(1) 020276 104411
2418 020300 170011
2419 020302 112737 000106 053616
2420 020310 012737 020342 001164
2421 020316 012700 020744
(1) 020322 172410
2422 020324 010037 001172
2423 020330 012700 021024
2424 020334 010037 001170
2425 020340 170001
2426 020342 172010
2427 020344 170205
2428 020346 010537 001166
2429 020352 170011
2430 020354 012700 020734
(1) 020360 174010
2431 020362 004537 004312
2432 020366 021024 020734
2433 020372 001407
2434 020374 012737 021024 001174
2435 020402 012737 020734 001176
2436 020410 104152
2437
2438
2439
2440 020412
(1) 020412 104411
2441 020414 170011
2442 020416 012737 020450 001164
2443 020424 012700 020774
(1) 020430 172410
2444 020432 010037 001172
2445 020436 012700 021014
2446 020442 010037 001170
2447 020446 170001
2448 020450 172010
2449 020452 170205
2450 020454 010537 001166
2451 020460 170011
2452 020462 012700 020734
(1) 020466 174010
2453 020470 004537 004312
2454 020474 021044 020734
2455 020500 001407
2456 020502 012737 021044 001174
2457 020510 012737 020734 001176

```
CCP7,CCSUM          ; EXP VS RECD.  
BEQ      CC13       ; BR IF OK.  
MOV      #CCP7,$TMP6  
MOV      #CCSUM,$TMP7  
ERROR    152        ; SUM INCORRECT.  
  
: NOW EXPONENT DIFF = 26 (32 OCT), MODE F.  
: ALIGNMENT NOT POSSIBLE.  
: CC13:  
LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.  
SETD  
MOVB      #'F,EM152X  
MOV      #CC14,@#$TMP2  
MOV      #CCP0,R0  
LDD      (R0),ACC   ;; LOAD ACO  
MOV      R0,$TMP5  
MOV      #CCP6,R0   ;FSRC  
MOV      R0,$TMP4  
SETF  
: SET FLOAT.  
CC14:  ADDF      (R0),ACO ;TEST INSTRUCTION  
STFPS    R5  
MOV      R5,$TMP3  
SETD  
: RETURN TO DOUBLE.  
MOV      #CCSUM,R0  
STD      ACO,(R0)   ;; STORE ACO  
JSR      R5,CHECK4  
CCP6,CCSUM  
BEQ      CC19       ; BR IF OK.  
MOV      #CCP6,$TMP6  
MOV      #CCSUM,$TMP7  
ERROR    152  
  
: EXPONENT DIFFERENCE=25 (31 OCT) MAX F MODE ALIGNMENT.  
: CC19:  
LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.  
SETD  
MOV      #CC20,@#$TMP2  
MOV      #CCP3,R0  
LDD      (R0),ACO   ;; LOAD ACO  
MOV      R0,$TMP5  
MOV      #CCP5,R0   ;FSRC  
MOV      R0,$TMP4  
SETF  
: FLOAT MODE.  
CC20:  ADDF      (R0),ACO ;TEST INSTRUCTION  
STFPS    R5  
MOV      R5,$TMP3  
SETD  
: REENTER DOUBLE MOVE  
MOV      #CCSUM,R0  
STD      ACO,(R0)   ;; STORE ACO  
JSR      R5,CHECK4  
CCP10,CCSUM  
BEQ      CC25       ; BR IF OK.  
MOV      #CCP10,$TMP6  
MOV      #CCSUM,$TMP7
```

2458 020516 104152
2459
2460
2461
2462
2463 020520
 (1) 020520 104411
2464 020522 112737 000104 053616
2465 020530 170011
2466 020532 012737 020562 001164
2467 020540 012700 020744
 (1) 020544 172410
2468 020546 010037 001172
2469 020552 012700 020774
2470 020556 010037 001170
2471 020562 172010
2472 020564 170205
2473 020566 010537 001166
2474 020572 012700 020734
 (1) 020576 174010
2475 020600 004537 004312
2476 020604 021054 020734
2477 020610 001407
2478 020612 012737 021054 001174
2479 020620 012737 020734 001176
2480 020626 104152
2481
2482
2483
2484
2485 020630
 (1) 020630 104411
2486 020632 170011
2487 020634 012737 020664 001164
2488 020642 012700 020744
 (1) 020646 172410
2489 020650 010037 001172
2490 020654 012700 021024
2491 020660 010037 001170
2492 020664 172010
2493 020666 170205
2494 020670 010537 001166
2495 020674 012700 020734
 (1) 020700 174010
2496 020702 004537 004312
2497 020706 021064 020734
2498 020712 001470
2499 020714 012737 021064 001174
2500 020722 012737 020734 001176
2501 020730 104152
2502 020732 000460
2503
2504 020734 000000 000000 000000
2505 020744 000200 000000 000000
2506 020754 016200 000000 000000
2507 020764 016400 000000 000000

ERROR 152
:
: NOW TRY A MINIMUM ALIGNMENT IN DOUBLE MODE.
: EXPONENT DIFF - 1.
:
CC25:
LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
MOVB #D,EM152X
SETD
MOV #CC26,@#STMP2
MOV #CCP0,R0
LDD (R0),AC0 ;; LOAD AC0
MOV R0,\$TMP5 ;; FSRC
MOV #CCP3,R0
MOV R0,\$TMP4
CC26: ADDD (R0),AC0 ;TEST INSTRUCTION
STFPS R5 ;GET FPS
MOV R5,\$TMP3
MOV #CCSUM,R0
STD AC0,(R0) ;; STORE AC0
JSR R5,CHECK4
CCP11,CCSUM
BEQ CC31 ; BR IF OK.
MOV #CCP11,\$TMP6
MOV #CCSUM,\$TMP7
ERROR 152
:
: FINALLY, ONE BETWEEN MAX AND MIN.
: EXPONENT DIFF - 25, DOUBLE MODE.
:
CC31:
LUPERR ;; LOOP HERE ON ERROR If SWR9 1.
SETD
MOV #CC32,@#STMP2
MOV #CCP0,R0
LDD (R0),AC0 ;; LOAD AC0
MOV R0,\$TMP5 ;; FSRC
MOV #CCP6,R0
MOV R0,\$TMP4
CC32: ADDD (R0),AC0 ;TEST INSTRUCTION
STFPS R5 ;GET FPS
MOV R5,\$TMP3
MOV #CCSUM,R0
STD AC0,(R0) ;; STORE AC0
JSR R5,CHECK4
CCP12,CCSUM
BEQ CCDONE
MOV #CCP12,\$TMP6
MOV #CCSUM,\$TMP7
ERROR 152
BR CCDONE
CCSUM: .FLT4 0
CCP0: 000200,0,0,0 ; EXP = 1
CCP1: 016200,0,0,0 ; EXP = 57. (71)
CCP2: 016400,0,0,0 ; EXP = 58. (72)

CJFPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJFPAA.P11 28-JAN-81 09:50 T32

MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-56
ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC)

SEQ 0086

```
2508 020774 000400 000000 000000 CCP3: 000400,0,0,0      : EXP = 2
2509 021004 031200 000000 000000 CCP4: 031200,0,0,0      : EXP = 101. (145)
2510 021014 006200 000000 000000 CCP5: 006200,0,0,0      : EXP = 25. (31)
2511 021024 006400 000000 000000 CCP6: 006400,0,0,0      : EXP = 26. (32)
2512
2513 021034 016200 000000 000000 CCP7: 016200,0,0,1      : DOUBLE SUM CCPO+CCP1
2514 021044 006200 000001 000000 CCP10: 006200,1,0,0      : SINGLE SUM CCPO+CCP5
2515 021054 000500 000000 000000 CCP11: 000500,0,0,0      : DOUBLE SUM CCPO+CCP3
2516 021064 006400 000000 040000 CCP12: 006400,0,40000,0 : DOUBLE SUM CCPO+CCP6
2517
2518 021074          CCDONE:
  (1) 021074 104412      CLRFPS      :: CLEAR FP STATUS...
  (3) 021076 000400      BR          TST33      ::...AND PROCEED.
```

2529
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 021100 000004
2530
2531
2532
2533 021102
(1) 021102 104411
2534 021104 170011
2535 021106 012737 021144 001164
2536 021114 112737 000104 053740
2537 021122 012700 022212
(1) 021126 172410
2538 021130 010037 001172
2539 021134 012700 022202
2540 021140 010037 001170
2541 021144 172010
2542 021146 170205
2543 021150 010537 001166
2544 021154 012700 022152
(1) 021160 174010
2545 021162 004537 004312
2546 021166 022212 022152
2547 021172 001407
2548 021174 012737 022212 001174
2549 021202 012737 022152 001176
2550 021210 104153
2551
2552 021212 012737 021242 001164
2553 021220 012700 022222
(1) 021224 172410
2554 021226 010037 001172
2555 021232 012700 022202
2556 021236 010037 001170
2557 021242 172010
2558 021244 170205
2559 021246 010537 001166
2560 021252 012700 022152
(1) 021256 174010
2561 021260 004537 004312
2562 021264 022222 022152
2563 021270 001407
2564 021272 012737 022222 001174
2565 021300 012737 022152 001176
2566 021306 104153
2567
2568
2569

```
*****  
*TEST 33      ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC)  
*  
*THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS  
*AND THE ALIGN FSRC EXPONENT ALGORITHM.  
*THIS IS THE SAME AS PRECEEDING TEST EXCEPT THAT  
*E(AC) IS GREATER THAN E(FSRC) SO THE FSRC OPERAND  
*IS THE C - TO BE ALIGNED.  
*****  
TST33: SCOPE  
: EXPONENT DIFF EQ OR GT 57. NO ALIGNMENT POSSIBLE.  
: BB1:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
SETD ; DOUBLE.  
MOV #BB2,$TMP2  
MOVB #'D,EM153X  
MOV #BBP2,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV R0,$TMP5 ;FSRC  
MOV #BBP1,R0  
MOV R0,$TMP4  
BB2: ADDD (R0),ACO ;TEST INSTRUCTION (EXP DIFF = 57.).  
STFPS R5  
MOV R5,$TMP3  
MOV #BBSUM,R0  
STD ACO,(R0) ;: STORE ACO  
JSR R5,CHECK4  
BBP2,BBSUM  
BEQ BB3  
MOV #BBP2,$TMP6  
MOV #BBSUM,$TMP7  
ERROR 153  
BB3: MOV #BB4,$TMP2  
MOV #BBP2A,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV R0,$TMP5 ;FSRC  
MOV #BBP1,R0  
MOV R0,$TMP4  
BB4: ADDD (R0),ACO ;TEST INSTRUCTION (EXP DIFF = 58.).  
STFPS R5  
MOV R5,$TMP3  
MOV #BBSUM,R0  
STD ACO,(R0) ;: STORE ACO  
JSR R5,CHECK4  
BBP2A,BBSUM  
BEQ BB6  
MOV #BBP2A,$TMP6  
MOV #BBSUM,$TMP7  
ERROR 153  
: NOW EXPONENT DIFF - 56., MAXIMUM DOUBLE ALIGNMENT.  
:
```


2619	021564	010537	001166		MOV	R5,\$TMP3	
2620	021570	170011			SETD		;RETURN DOUBLE MODE.
2621	021572	012700	022152		MOV	#BBSUM,R0	
(1)	021576	174010			STD	AC0,(R0)	:: STORE ACO
2622	021600	004537	004312		JSR	R5,CHECK4	
2623	021604	022172	022152		BBPOA,BBSUM		
2624	021610	001407			BEQ	BB20	; BR IF OK.
2625	021612	012737	022172	001174	MOV	#BBPOA,\$TMP6	
2626	021620	012737	022152	001176	MOV	#BBSUM,\$TMP7	
2627	021626	104153			ERROR	153	
2628							
2629							
2630							
2631	021630						
(1)	021630	104411			BB20:	LUPERR	:: LOOP HERE ON ERROR IF SWR9 - 1.
2632	021632	170011			SETD		
2633	021634	012737	021666	001164	MOV	#BB21,@\$TMP2	
2634	021642	012700	022232		MOV	#BBP3,R0	
(1)	021646	172410			LDD	(R0),AC0	:: LOAD ACO
2635	021650	010037	001172		MOV	R0,\$TMP5	
2636	021654	012700	022202		MOV	#BBP1,R0	;FSRC
2637	021660	010037	001170		MOV	R0,\$TMP4	
2638	021664	170001			SETF		; FLOAT.
2639	021666	172010			BB21:	ADDF	(R0),AC0
2640	021670	170205			STFPS	R5	;TEST INSTRUCTION
2641	021672	010537	001166		MOV	R5,\$TMP3	
2642	021676	170011			SETD		;RETURN DOUBLE MODF
2643	021700	012700	022152		MOV	#BBSUM,R0	
(1)	021704	174010			STD	AC0,(R0)	:: STORE ACO
2644	021706	004537	004312		JSR	R5,CHECK4	
2645	021712	022262	022152		BBP7,BBSUM		
2646	021716	001407			BEQ	BB26	
2647	021720	012737	022262	001174	MOV	#BBP7,\$TMP6	
2648	021726	012737	022152	001176	MOV	#BBSUM,\$TMP7	
2649	021734	104153			ERROR	153	
2650							
2651							
2652							
2653	021736						
(1)	021736	104411			BB26:	LUPERR	:: LOOP HERE ON ERROR IF SWR9 1.
2654	021740	170011			SETD		
2655	021742	012737	022000	001164	MOV	#BB27,@\$TMP2	
2656	021750	112737	000104	053740	MOVB	#'D,EM153X	
2657	021756	012700	022252		MOV	#BBP5,R0	
(1)	021762	172410			LDD	(R0),AC0	:: LOAD ACO
2658	021764	010037	001172		MOV	R0,\$TMP5	
2659	021770	012700	022202		MOV	#BBP1,R0	;FSRC
2660	021774	010037	001170		MOV	R0,\$TMP4	
2661	022000	172010			BB27:	ADDD	(R0),AC0
2662	022002	170205			STFPS	R5	;TEST INSTRUCTION
2663	022004	010537	001166		MOV	R5,\$TMP3	
2664	022010	012700	022152		MOV	#BBSUM,R0	
(1)	022014	174010			STD	AC0,(R0)	:: STORE ACO
2665	022016	004537	004312		JSR	R5,CHECK4	
2666	022022	022302	022152		BBP11,BBSUM		
2667	022026	001407			BEQ	BB32	; BR IF OK.

```
2668 022030 012737 022302 001174      MOV      #BBP11,$TMP6
2669 022036 012737 022152 001176      MOV      #BBSUM,$TMP7
2670 022044 104153                      ERROR    153
2671
2672      ; NOW ONE IN THE MIDDLE, DIFF = 25, DOUBLE MODE.
2673
2674      BB32:
(1) 022046 104411      LUPERR                      ;; LOOP HERE ON ERROR IF SWR9 = 1.
2675 022050 170011      SETD
2676 022052 012737 022102 001164      MOV      #BB33,@#$TMP2
2677 022060 012700 022162      MOV      #BBP0,R0
(1) 022064 172410      LDD      (R0),AC0          ;; LOAD AC0
2678 022066 010037 001172      MOV      R0,$TMP5
2679 022072 012700 022202      MOV      #BBP1,R0          ;FSRC
2680 022076 010037 001170      MOV      R0,$TMP4
2681 022102 172010      BB33:  ADDD      (R0),AC0    ;TEST INSTRUCTION
2682 022104 170205      STFPS   R5
2683 022106 010537 001166      MOV      R5,$TMP3
2684 022112 012700 022152      MOV      #BBSUM,R0
(1) 022116 174010      STD      AC0,(R0)          ;; STORE AC0
2685 022120 004537 004312      JSR      R5,CHECK4
2686 022124 022312 022152      BBP12,BBSUM
2687 022130 001474      BEQ      BBDONE           ; BR IF OK.
2688 022132 012737 022312 001174      MOV      #BBP12,$TMP6
2689 022140 012737 022152 001176      MOV      #BBSUM,$TMP7
2690 022146 104153                      ERROR    153
2691 022150 000464                      BR       BBDONE
2692
2693 022152 000000 000000 000000  BBSUM:  .FLT4  0           ; RECEIVED SUM.
2694 022162 006400 000000 000000  BBP0:   006400,0,0,0     ; EXP = 26. (32)
2695 022172 006600 000000 000000  BBP0A: 006600,0,0,0     ; EXP = 27.
2696 022202 000200 000000 000000  BBP1:   000200,0,0,0     ; EXP = 1.
2697 022212 016400 000000 000000  BBP2:   016400,0,0,0     ; EXP = 58. (72)
2698 022222 016600 000000 000000  BBP2A:  016600,0,0,0     ; EXP = 59.
2699 022232 006200 000000 000000  BBP3:   006200,0,0,0     ; EXP = 25. (31)
2700 022242 016200 000000 000000  BBP4:   016200,0,0,0     ; EXP = 57. (71)
2701 022252 000400 000000 000000  BBP5:   000400,0,0,0     ; EXP = 2.
2702
2703 022262 006200 000001 000000  BBP7:   006200,1,0,0     ; SINGLE SUM 3+1
2704 022272 016200 000000 000000  BBP10:  016200,0,0,1     ; DOUBLE SUM 4+1
2705 022302 000500 000000 000000  BBP11:  000500,0,0,0     ; DOUBLE SUM 5+1
2706 022312 006400 000000 040000  BBP12:  006400,0,4000,0  ; DOUBLE SUM 1+0
2707
2708 022322      BBDONE:
(1) 022322 104412      CLRFPS
(3) 022324 000400      BR       TST34           ;; CLEAR FP STATUS...
                          ;...AND PROCEED.
```

```

2716
(3)
(4)
(4)
(4)
(4)
(3)
(2) 022326 000004
2717
2718
2719
2720 022330
(1) 022330 104411
2721 022332 012737 023204 001176
2722 022340 172011
2723 022342 012737 022372 001164
2724 022350 012700 023224
(1) 022354 172410
2725 022356 010037 001172
2726 022362 012700 023224
2727 022366 010037 001170
2728 022372 172010
2729 022374 170205
2730 022376 010537 001166
2731 022402 012700 023204
(1) 022406 174010
2732 022410 004537 004312
2733 022414 023324 023204
2734 022420 001404
2735 022422 012737 023224 001174
2736 022430 104154
2737
2738
2739
2740 022432
(1) 022432 104411
2741 022434 170011
2742 022436 012737 022466 001164
2743 022444 012700 023234
(1) 022450 172410
2744 022452 010037 001172
2745 022456 012700 023224
2746 022462 010037 001170
2747 022466 172010
2748 022470 170205
2749 022472 010537 001166
2750 022476 012700 023204
(1) 022502 174010
2751 022504 004537 004312
2752 022510 023214 023204
2753 022514 001404
2754 022516 012737 023214 001174
2755 022524 104154
2756
2757
2758
    
```

```

*****
: TEST 34      ADD WITH POSITIVE AND NEGATIVE OPERANDS
:
: THIS IS A TEST OF THE ADD INSTRUCTION USING BOTH
: POSITIVE AND NEGATIVE OPERANDS.
:
: *****
TST34: SCOPE
:
: FIRST, BOTH OPERANDS NEGATIVE
:
DD1:
LUPERR                ;; LOOP HERE ON ERROR IF SWR9 = 1.
MOV #DDSUM,$TMP7      ;; SET RECEIVED POINTER.
SETD                  ;; DOUBLE MODE.
MOV #DD2,@$TMP2
MOV #DDP1,R0
LDD (R0),AC0          ;; LOAD ACO
MOV R0,$TMP5          ;; ESRC
MOV #DDP1,R0
MOV R0,$TMP4
DD2: ADDD (R0),AC0     ;; TEST INSTRUCTION
STFPS R5              ;; GET FPS
MOV R5,$TMP3
MOV #DDSUM,R0
STD ACO,(R0)          ;; STORE ACO
JSR R5,CHECK4
DDP9,DDSUM
BEQ DD7               ;; BR IF CORRECT.
MOV #DDP9,$TMP6
ERROR 154

: AC POS, SRC NEG, EQUAL MAGNITUDE.
:
DD7:
LUPERR                ;; LOOP HERE ON ERROR IF SWR9 = 1.
SETD
MOV #DD8,@$TMP2
MOV #DDP2,R0
LDD (R0),AC0          ;; LOAD ACO
MOV R0,$TMP5          ;; FSPC
MOV #DDP1,R0
MOV R0,$TMP4
DD8: ADDD (R0),AC0     ;; TEST INSTRUCTION
STFPS R5              ;; GET FPS
MOV R5,$TMP3
MOV #DDSUM,R0
STD ACO,(R0)          ;; STORE ACO
JSR R5,CHECK4
DDP0,DDSUM
BEQ DD12              ;; BR IF OK.
MOV #DDP0,$TMP6
ERROR 154

: AC NEG, SRC POS, EQUAL MAGNITUDE.
:
    
```

```

2759 022526          DD12:
(1) 022526 104411      LUPERR                ;; LOOP HERE ON ERROR IF SWR9 - 1.
2760 022530 170011      SETD
2761 022532 012737 022562 001164  MOV #DD13,@#$TMP2
2762 022540 012700 023224  MOV #DDP1,R0
(1) 022544 172410      LDD (R0),AC0          ;; LOAD AC0
2763 022546 010037 001172  MOV R0,$TMP5
2764 022552 012700 023234  MOV #DDP2,R0          ;FSRC
2765 022556 010037 001170  MOV R0,$TMP4
2766 022562 172010      DD13: ADDD (R0),AC0      ;TEST INSTRUCTION
2767 022564 170205      STFPS R5              ;GET FPS
2768 022566 010537 001166  MOV R5,$TMP3
2769 022572 012700 023204  MOV #DDSUM,R0
(1) 022576 174010      STD AC0,(R0)          ;; STORE AC0
2770 022600 004537 004312  JSR R5,CHECK4
2771 022604 023214 023204  DDP0,DDSUM
2772 022610 001404      BEQ DD16              ; BR IF OK.
2773 022612 012737 023214 001174  MOV #DDP0,$TMP6
2774 022620 104154      ERROR 154
2775
2776      ; AC POS, SRC NEG, AC > SRC
2777
2778 022622          DD16:
(1) 022622 104411      LUPERR                ;; LOOP HERE ON ERROR IF SWR9 1.
2779 022624 170011      SETD
2780 022626 012737 022656 001164  MOV #DD17,@#$TMP2
2781 022634 012700 023244  MOV #DDP3,R0
(1) 022640 172410      LDD (R0),AC0          ;; LOAD AC0
2782 022642 010037 001172  MOV R0,$TMP5
2783 022646 012700 023274  MOV #DDP6,R0          ;ESPC
2784 022652 010037 001170  MOV R0,$TMP4
2785 022656 172010      DD17: ADDD (R0),AC0      ;TEST INSTRUCTION
2786 022660 170205      STFPS R5              ;GET FPS
2787 022662 010537 001166  MOV R5,$TMP3
2788 022666 012700 023204  MOV #DDSUM,R0
(1) 022672 174010      STD AC0,(R0)          ;; STORE AC0
2789 022674 004537 004312  JSR R5,CHECK4
2790 022700 023304 023204  DDP7,DDSUM
2791 022704 001404      BEQ DD22              ; BR IF OK.
2792 022706 012737 023304 001174  MOV #DDP7,$TMP6
2793 022714 104154      ERROR 154
2794
2795      ; AC NEG, SRC POS, AC < SRC
2796
2797 022716          DD22:
(1) 022716 104411      LUPERR                ;; LOOP HERE ON ERROR IF SWR9 - 1.
2798 022720 170011      SETD
2799 022722 012737 022752 001164  MOV #DD23,@#$TMP2
2800 022730 012700 023274  MOV #DDP6,R0
(1) 022734 172410      LDD (R0),AC0          ;; LOAD AC0
2801 022736 010037 001172  MOV R0,$TMP5
2802 022742 012700 023244  MOV #DDP3,R0          ;FSPC
2803 022746 010037 001170  MOV R0,$TMP4
2804 022752 172010      DD23: ADDD (R0),AC0      ;TEST INSTRUCTION
2805 022754 170205      STFPS R5              ;GET FPS
2806 022756 010537 001166  MOV R5,$TMP3

```

```

2807 022762 012700 023204      MOV    #DDSUM,R0
      (1) 022766 174010      STD    ACO,(R0)          ;; STORE ACO
2808 022770 004537 004312      JSR    R5,CHECK4
2809 022774 023304 023204      DDP7,DDSUM
2810 023000 001404      BEQ    DD30              ; BR IF OK.
2811 023002 012737 023304 001174  MOV    #DDP7,$TMP6
2812 023010 104154      ERROR  154
2813
2814      ; AC POS, SRC NEG, AC < SRC
2815      ;
2816 023012      DD30:
      (1) 023012 104411      LUPERR                    ;; LOOP HERE ON ERROR IF SWR9 - 1.
2817 023014 170011      SETD
2818 023016 012737 023046 001164  MOV    #DD31,@#$TMP2
2819 023024 012700 023254      MOV    #DDP4,R0
      (1) 023030 172410      LDD    (R0),ACO          ;; LOAD ACO
2820 023032 010037 001172      MOV    R0,$TMP5
2821 023036 012700 023264      MOV    #DDP5,R0          ;FSRC
2822 023042 010037 001170      MOV    R0,$TMP4
2823 023046 172010      DD31: ADDD    (R0),ACO      ;TEST INSTRUCTION
2824 023050 170205      STFPS  R5                ;GET FPS
2825 023052 010537 001166      MOV    R5,$TMP3
2826 023056 012700 023204      MOV    #DDSUM,R0
      (1) 023062 174010      STD    ACO,(R0)          ;; STORE ACO
2827 023064 004537 004312      JSR    R5,CHECK4
2828 023070 023314 023204      DDP8,DDSUM
2829 023074 001404      BEQ    DD36              ; BR IF OK.
2830 023076 012737 023314 001174  MOV    #DDP8,$TMP6
2831 023104 104154      ERROR  154
2832
2833      ; AC NEG, SRC POS, AC > SRC
2834      ;
2835 023106      DD36:
      (1) 023106 104411      LUPERR                    ;; LOOP HERE ON ERROR IF SWR9 - 1.
2836 023110 170011      SETD
2837 023112 012737 023142 001164  MOV    #DD37,@#$TMP2
2838 023120 012700 023264      MOV    #DDP5,R0
      (1) 023124 172410      LDD    (R0),ACO          ;; LOAD ACO
2839 023126 010037 001172      MOV    R0,$TMP5
2840 023132 012700 023254      MOV    #DDP4,R0          ;FSRC
2841 023136 010037 001170      MOV    R0,$TMP4
2842 023142 172010      DD37: ADDD    (R0),ACO      ;TEST INSTRUCTION
2843 023144 170205      STFPS  R5                ;GET FPS
2844 023146 010537 001166      MOV    R5,$TMP3
2845 023152 012700 023204      MOV    #DDSUM,R0
      (1) 023156 174010      STD    ACO,(R0)          ;; STORE ACO
2846 023160 004537 004312      JSR    R5,CHECK4
2847 023164 023314 023204      DDP8,DDSUM
2848 023170 001404      BEQ    1$                ; BR IF OK.
2849 023172 012737 023314 00 174  MOV    #DDP8,$TMP6
2850 023200 104154      ERROR  154
2851 023202 000454      1$: BR    DDDONE
2852
2853 023204 000000 000000 000000 000000 DDSUM: .FLT4  0          ; RECEIVED SUM.
2854 023214 000000 000000 000000 000000 DDP0:  .FLT4  0          ; FLOATING 0.
2855 023224 100200 000000 000000 000000 DDP1:  100200,0,0,0     ; -P2.

```

2856	023234	000200	000000	000000	DDP2:	000200,0,0,0	
2857	023244	001100	000000	000000	DDP3:	001100,0,0,0	
2858	023254	000600	000000	000000	DDP4:	000600,0,0,0	
2859	023264	101100	000000	000000	DDP5:	101100,0,0,0	: -P3
2860	023274	100600	000000	000000	DDP6:	100600,0,0,0	: -P4
2861	023304	001000	000000	000000	DDP7:	001000,0,0,0	: SUM P3+P6
2862	023314	101000	000000	000000	DDP8:	101000,0,0,0	: SUM F5+P4
2863	023324	100400	000000	000000	DDP9:	100400,0,0,0	: SUM P1+P1
2864							
2865	023334				DDDONE:		
(1)	023334	104412			CLRFPS		:: CLEAR FP STATUS...
(3)	023336	000400			BR TST35		::...AND PROCEED.

```
2873 *****  
(3) *TEST 35 SUBD WITH POSITIVE AND NEGATIVE OPERANDS  
(4) *  
(4) * THIS IS A TEST OF THE SUBD INSTRUCTION USING VARIOUS OPERANDS..  
(4) *  
(3) *****  
(2) 023340 000004 TST35: SCOPE  
2874 : N - N = 0  
2875 :  
2876 :  
2877 EE1: LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.  
(1) 023342 104411 MOV #EEDIFF,$TMP7 ; RECEIVED DIFF POINTER.  
2878 023344 012737 023732 001176 SETD ; DOUBLE MODE.  
2879 023352 170011 MOV #1$,@$TMP2  
2880 023354 012737 023404 001164 MOV #EEP1,R0  
2881 023362 012700 023742 LDD (R0),AC0 ;; LOAD ACO  
(1) 023366 172410 MOV R0,$TMP5 ;FSPC  
2882 023370 010037 001172 MOV #EEP1,R0  
2883 023374 012700 023742 MOV R0,$TMP4  
2884 023400 010037 001170 1$: SUBD (R0),AC0 ;TEST INSTRUCTION  
2885 023404 173010 STFPS R5 ;GET FPS  
2886 023406 170205 MOV R5,$TMP3  
2887 023410 010537 001166 MOV #EEDIFF,R0  
2888 023414 012700 023732 STD ACO,(R0) ;; STORE ACO  
(1) 023420 174010 JSR R5,CHECK4  
2889 023422 004537 004312 EEP3,EEDIFF ; EXP VS RECD.  
2890 023426 023762 023732 BEQ EE2 ; BR IF OK.  
2891 023432 001404 MOV #EEP3,$TMP6  
2892 023434 012737 023762 001174 ERROR 155  
2893 023442 104155  
2894 :  
2895 : (-N) - (-N) = 0  
2896 :  
2897 EE2: LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.  
(1) 023444 104411 SETD  
2898 023446 170011 MOV #1$,@$TMP2  
2899 023450 012737 023500 001164 MOV #EEP2,R0  
2900 023456 012700 023752 LDD (R0),AC0 ;; LOAD ACO  
(1) 023462 172410 MOV R0,$TMP5 ;FSPC  
2901 023464 010037 001172 MOV #EEP2,R0  
2902 023470 012700 023752 MOV R0,$TMP4  
2903 023474 010037 001170 1$: SUBD (R0),AC0 ;TEST INSTRUCTION  
2904 023500 173010 STFPS R5 ;GET FPS  
2905 023502 170205 MOV R5,$TMP3  
2906 023504 010537 001166 MOV #EEDIFF,R0  
2907 023510 012700 023732 STD ACO,(R0) ;; STORE ACO  
(1) 023514 174010 JSR R5,CHECK4  
2908 023516 004537 004312 EEP3,EEDIFF  
2909 023522 023762 023732 BEQ EE3 ; BR IF OK  
2910 023526 001404 MOV #EEP3,$TMP6  
2911 023530 012737 023762 001174 ERROR 155  
2912 023536 104155  
2913 :  
2914 : NOW A MIXED NUMBER MINUS A WHOLE NUMBER.  
2915 :
```



```

2916 023540          EE3:      LUPERR                ;; LOOP HERE ON ERROR IF SWR9 = 1.
(1) 023540 104411      SETD
2917 023542 170011      MOV      #1$,@#STMP2
2918 023544 012737 023574 001164  MOV      #EEP4,R0
2919 023552 012700 023772      LDD      (R0),AC0      ;; LOAD AC0
(1) 023556 172410      MOV      R0,$TMP5
2920 023560 010037 001172      MOV      #EEP5,R0      ;FSRC
2921 023564 012700 024002      MOV      R0,$TMP4
2922 023570 010037 001170      SUBD     (R0),AC0      ;TEST INSTRUCTION
2923 023574 173010      STFPS    R5            ;GET FPS
2924 023576 170205      MOV      R5,$TMP3
2925 023600 010537 001166      MOV      #EEDIFF,R0
2926 023604 012700 023732      STD      AC0,(R0)      ;; STORE AC0
(1) 023610 174010      JSR      R5,CHECK4
2927 023612 004537 004312      EEP6,EEDIFF
2928 023616 024012 023732      BEQ      EE4            ; BR IF OK
2929 023622 001404
2930 023624 012737 024012 001174  MOV      #EEP6,$TMP6
2931 023632 104155      ERROR    155
2932
2933      ; NOW A WHOLE NUMBER MINUS A MIXED NUMBER.
2934
2935 023634          EE4:      LUPERR                ;; LOOP HERE ON ERROR IF SWR9 = 1.
(1) 023634 104411      SETD
2936 023636 170011      MOV      #1$,@#STMP2
2937 023640 012737 023670 001164  MOV      #EEP5,R0
2938 023646 012700 024002      LDD      (R0),AC0      ;; LOAD AC0
(1) 023652 172410      MOV      R0,$TMP5
2939 023654 010037 001172      MOV      #EEP4,R0      ;FSRC
2940 023660 012700 023772      MOV      R0,$TMP4
2941 023664 010037 001170      SUBD     (R0),AC0      ;TEST INSTRUCTION
2942 023670 173010      STFPS    R5            ;GET FPS
2943 023672 170205      MOV      R5,$TMP3
2944 023674 010537 001166      MOV      #EEDIFF,R0
2945 023700 012700 023732      STD      AC0,(R0)      ;; STORE AC0
(1) 023704 174010      JSR      R5,CHECK4
2946 023706 004537 004312      EEP7,EEDIFF
2947 023712 024022 023732      BEQ      EE5            ; BR IF OK
2948 023716 001404
2949 023720 012737 024022 001174  MOV      #EEP7,$TMP6
2950 023726 104155      ERROR    155
2951 023730 000440          EE5:      BR      EEDONE
2952
2953 023732 000000 000000 000000  EEDIFF: .FLT4 0      ; RECEIVED DIFFERENCE.
2954 023742 000200 000001 000001  EEP1: 000200,1,1,1
2955 023752 100200 000001 000001  EEP2: 100200,1,1,1
2956 023762 000000 000000 000000  EEP3: 0,0,0,0      ; P1-P1 AND P2-P2
2957 023772 040600 144724 000000  EEP4: 040600,144724,0,0 ; MIXED NUMBER.
2958 024002 040600 000000 000000  EEP5: 040600,0,0,0    ; WHOLE NUMBER.
2959 024012 036711 152000 000000  EEP6: 036711,152000,0,0 ; P4-P5 (MIXED-WHOLE).
2960 024022 136711 152000 000000  EEP7: 136711,152000,0,0 ; P5-P4 (WHOLE-MIXED).
2961
2962 024032          EEDONE:
(1) 024032 104412      CLRFPs                ;; CLEAR FP STATUS...
(3) 024034 000400      BR      TST36         ;...AND PROCEED.

```

2974
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 024036 000004
2975
2976
2977
2978 024040 012700 054174
2979 024044 112720 000101
2980 024050 112720 000104
2981 024054 112720 000104
2982 024060 104411
2983 024062 170011
2984 024064 012737 024470 001176
2985 024072 012737 024122 001164
2986 024100 012700 024520
(1) 024104 172410
2987 024106 010037 001172
2988 024112 012700 024530
2989 024116 010037 001170
2990 024122 172010
2991 024124 170205
2992 024126 010537 001166
2993 024132 012700 024470
(1) 024136 174010
2994 024140 004537 004312
2995 024144 024540 024470
2996 024150 001404
2997 024152 012737 024540 001174
2998 024160 104156
2999
3000
3001
3002 024162
(1) 024162 104411
3003 024164 170011
3004 024166 012737 024216 001164
3005 024174 012700 024500
(1) 024200 172410
3006 024202 010037 001172
3007 024206 012700 024510
3008 024212 010037 001170
3009 024216 172010
3010 024220 170205
3011 024222 010537 001166
3012 024226 012700 024470
(1) 024232 174010
3013 024234 004537 004312

```
*****  
*TEST 36 TEST NORMALIZE FLOW USING ADD AND SUBD  
*  
* TEST THAT THE RESULTS OF AN ADD OR SUBTRACT ARE  
* PROPERLY NORMALIZED IN ACO.  
* DATA IS SELECTED TO TEST TWO CASES.  
* FIRST THE MINIMUM SITUATION REQUIRING ONE  
* LEFT SHIFT AND THEN THE MAXIMUM SITUATION  
* REQUIRING 56 SHIFTS.  
*****  
TST36: SCOPE  
: MIN CASE ADDING -- RESULT REQUIRES 1 SHIFT TO NORMALIZE.  
: :  
FF1: MOV #EM156X,R0  
MOV #A,(R0)+  
MOVB #D,(R0)+  
MOVB #D,(R0)+ ; PRIME ERROR TEXT ADD.  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
SETD ; DOUBLE MODE.  
MOV #FFSUM,$TMP7  
MOV #1$,@$TMP2  
MOV #FFP2,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV R0,$TMP5 ;: FSPC  
MOV #FFP3,R0  
MOV R0,$TMP4  
1$: ADDD (R0),ACO ;: TEST INSTRUCTION  
STFPS R5 ;: GET FPS  
MOV R5,$TMP3  
MOV #FFSUM,R0  
STD ACO,(R0) ;: STORE ACO  
JSR R5,CHECK4  
FFP4,FFSUM  
BEQ FF2 ; BR IF OK.  
MOV #FFP4,$TMP6  
ERROR 156  
: :  
: MAX CASE ADDING -- RESULT REQUIRES 56 SHIFTS TO NORMALIZE.  
: :  
FF2: LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
SETD  
MOV #1$,@$TMP2  
MOV #FFP0,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV R0,$TMP5 ;: FSRC  
MOV #FFP1,R0  
MOV R0,$TMP4  
1$: ADDD (R0),ACO ;: TEST INSTRUCTION  
STFPS R5 ;: GET FPS  
MOV R5,$TMP3  
MOV #FFSUM,R0  
STD ACO,(R0) ;: STORE ACO  
JSR R5,CHECK4
```

```

3014 024240 024540 024470      FFP4,FFSUM
3015 024244 001404      BEQ      FF3      ; BR IF OK.
3016 024246 012737 024540 001174      MOV      #FFP4,$TMP6
3017 024254 104156      ERROR    156
3018
3019      ; MIN CASE -- SUBTRACTING.
3020
3021 024256 012700 054174      FF3::  MOV      #EM156X,R0
3022 024262 112720 000123      MOVB     #'S,(R0)+
3023 024266 112720 000125      MOVB     #'U,(R0)+
3024 024272 112720 000102      MOVB     #'B,(R0)+      ; PRIME ERROR TEXT = SUB
3025 024276 104411      LUPERR
3026 024300 170011      SETD
3027 024302 012737 024332 001164      MOV      #1$,@$TMP2
3028 024310 012700 024520      MOV      #FFP2,R0
(1) 024314 172410      LDD      (R0),AC0      ;; LOAD AC0
3029 024316 010037 001172      MOV      R0,$TMP5
3030 024322 012700 024560      MOV      #FFP6,R0      ;FSRC
3031 024326 010037 001170      MOV      R0,$TMP4
3032 024332 173010      1$:  SUBD     (R0),AC0      ;TEST INSTRUCTION
3033 024334 170205      STFPS    R5           ;GET FPS
3034 024336 010537 001166      MOV      R5,$TMP3
3035 024342 012700 024470      MOV      #FFSUM,R0
(1) 024346 174010      STD      AC0,(R0)      ;; STORE AC0
3036 024350 004537 004312      JSR      R5,CHECK4
3037 024354 024540 024470      FFP4,FFSUM
3038 024360 001404      BEQ      FF4      ; BR IF OK.
3039 024362 012737 024540 001174      MOV      #FFP4,$TMP6
3040 024370 104156      ERROR    156
3041
3042      ; MAX CASE -- SUBTRACTING.
3043
3044 024372      FF4:
(1) 024372 104411      LUPERR
3045 024374 170011      SETD
3046 024376 012737 024426 001164      MOV      #1$,@$TMP2
3047 024404 012700 024500      MOV      #FFP0,R0
(1) 024410 172410      LDD      (R0),AC0      ;; LOAD AC0
3048 024412 010037 001172      MOV      R0,$TMP5
3049 024416 012700 024550      MOV      #FFP5,R0      ;FSRC
3050 024422 010037 001170      MOV      R0,$TMP4
3051 024426 173010      1$:  SUBD     (R0),AC0      ;TEST INSTRUCTION
3052 024430 170205      STFPS    R5           ;GET FPS
3053 024432 010537 001166      MOV      R5,$TMP3
3054 024436 012700 024470      MOV      #FFSUM,R0
(1) 024442 174010      STD      AC0,(R0)      ;; STORE AC0
3055 024444 004537 004312      JSR      R5,CHECK4
3056 024450 024540 024470      FFP4,FFSUM
3057 024454 001404      BEQ      FF5      ; BR IF OK.
3058 024456 012737 024540 001174      MOV      #FFP4,$TMP6
3059 024464 104156      ERROR    156
3060
3061 024466 000440      FF5:  BR      FFDONE
3062
3063 024470 000000 000000 000000 000000  FFSUM:  .FLT4  0      ; RECEIVED SUM AND DIFF.
3064 024500 016000 000000 00G000  FFP0:  016000,0,0,1  ; EXP 70

```

3065	024510	116000	000000	000000	FFP1:	116000,0,0,0	:	EXP 70 AND NEG SIGN.
3066	024520	000500	000000	000000	FFP2:	000500,0,0,0	:	EXP 2
3067	024530	100400	000000	000000	FFP3:	100400,0,0,0	:	EXP 2 AND NEG SIGN.
3068	024540	000200	000000	000000	FFP4:	00200,0,0,0	:	P0+P1 AND P2+P3...
3069							:	...ALSO P0-P5 AND P2-P6.
3070	024550	016000	000000	000000	FFP5:	016000,0,0,0	:	NEG OF P1.
3071	024560	000400	000000	000000	FFP6:	000400,0,0,0	:	NEG CF P3.
3072								
3073	024570				FFDONE:			
(1)	024570	104412			CLRFPS		::	CLEAR FP STATUS...
(3)	024572	000400			BR TST37		::	...AND PROCEED.

```
3085 .....  
(3) *TEST 37 TEST ROUND\TRUNCATE USING ADDD  
(4) *  
(4) * THIS IS A TEST OF THE ROUND AND TRUNCATE FLOWS.  
(4) * IN PARTICULAR TWO THINGS ARE TESTED:  
(4) * FIRST A CONDITION IN WHICH ROUNDING RESULTS  
(4) * IN THE NEED FOR RENORMALIZATION.  
(4) * AND SECOND, THAT THE FPS CONDITION CODES (N AND Z BITS)  
(4) * ARE CORRECTLY SET IN VARIOUS COMBINATIONS.  
(4) *  
(3) .....  
(2) 024574 000004  
3086 024576  
(1) 024576 104411  
3087 024600 012704 000200  
3088 024604 170104  
3089 024606 012737 024626 001164  
3090 024614 012700 025456  
(1) 024620 172410  
3091 024622 012700 025466  
3092 024626 172010  
3093 024630 170205  
3094 024632 012700 025446  
(1) 024636 174010  
3095 024640 020405  
3096 024642 001005  
3097 024644 004537 004312  
3098 024650 025476 025446  
3099 024654 001421  
3100  
3101 024656 010537 001166  
3102 024662 010437 001170  
3103 024666 012737 025466 001172  
3104 024674 012737 025456 001174  
3105 024702 012737 025476 001176  
3106 024710 012737 025446 001200  
3107 024716 104160  
3108  
3109  
3110  
3111  
3112 024720  
(1) 024720 104411  
3113 024722 012704 000200  
3114 024726 170104  
3115 024730 012737 024750 001164  
3116 024736 012700 025526  
(1) 024742 172410  
3117 024744 012700 025536  
3118 024750 172010  
3119 024752 170205  
3120 024754 012700 025446  
(1) 024760 174010  
3121 024762 052704 000004  
3122 024766 020405  
3123 024770 001005
```

```
.....  
TST37: SCOPE  
HH1: LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.  
MOV #200,R4  
LDFPS R4  
MOV #HH2,@$TMP2  
MOV #HHP0,R0  
LDD (R0),ACO ;; LOAD ACO  
MOV #HHP1,R0 ;; SRC OPERAND.  
ADDD (R0),ACO ;; TEST INSTRUCTION  
STFPS R5 ;; GET FPS  
MOV #HHSUM,R0  
STD ACO,(R0) ;; STORE ACO  
CMP R4,R5 ;; FPS RIGHT ??  
BNE 1$ ;; NO.  
JSR R5,CHECK4 ;; YES, DATA CORRECT ??  
HHP2,HHSUM  
BEQ HH7 ;; BR IF SO.  
1$: MOV R5,$TMP3  
MOV R4,$TMP4  
MOV #HHP1,$TMP5  
MOV #HHP0,$TMP6  
MOV #HHP2,$TMP7  
MOV #HHSUM,$TMP10  
ERROR 160 ;; SUM OR FPS BAD.  
.....  
; TEST THAT NORMALIZE CAN PRODUCE A ZERO EXPONENT.  
; AND THAT THE FPS Z BIT GETS SET.  
.....  
HH7: LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.  
MOV #200,R4  
LDFPS R4  
MOV #HH8,@$TMP2  
MOV #HHP5,R0  
LDD (R0),ACO ;; LOAD ACO  
MOV #HHP6,R0 ;; SRC OPERAND.  
ADDD (R0),ACO ;; TEST INSTRUCTION  
STFPS R5 ;; GET FPS  
MOV #HHSUM,R0  
STD ACO,(R0) ;; STORE ACO  
BIS #4,R4 ;; SET EXP FPS.  
CMP R4,R5 ;; FPS RIGHT ??  
BNE 1$ ;; NO.
```

3124 024772 004537 004312
3125 024776 025516 025446
3126 025002 001421
3127
3128 025004 010537 001166
3129 025010 010437 001170
3130 025014 012737 025536 001172
3131 025022 012737 025526 001174
3132 025030 012737 025516 001176
3133 025036 012737 025446 001200
3134 025044 104160
3135
3136
3137
3138
3139
3140 025046
(1) 025046 104411
3141 025050 012704 042200
3142 025054 170104
3143 025056 012737 025076 001164
3144 025064 012700 025556
(1) 025070 172410
3145 025072 012700 025566
3146 025076 172010
3147 025100 170205
3148 025102 012700 025446
(1) 025106 174010
3149 025110 052704 100014
3150 025114 020405
3151 025116 001005
3152 025120 004537 004312
3153 025124 025546 025446
3154 025130 001421
3155
3156 025132 010537 001166
3157 025136 010437 001170
3158 025142 012737 025566 001172
3159 025150 012737 025556 001174
3160 025156 012737 025546 001176
3161 025164 012737 025446 001200
3162 025172 104160
3163
3164
3165
3166 025174
(1) 025174 104411
3167 025176 012704 000200
3168 025202 170104
3169 025204 012737 025224 001164
3170 025212 012700 025556
(1) 025216 172410
3171 025220 012700 025556
3172 025224 172010
3173 025226 170205
3174 025230 012700 025446

```
JSR R5,CHECK4 ; YES, DATA RIGHT ??  
HHP4,HHSUM  
BEQ HH1 ; BR IF SO.  
  
1$: MOV R5,$TMP3  
MOV R4,$TMP4  
MOV #HHP6,$TMP5  
MOV #HHP5,$TMP6  
MOV #HHP4,$TMP7  
MOV #HHSUM,$TMP10  
ERROR 160 ; SUM OR FPS BAD.  
  
: THIS IS A TEST OF THE R/T ALGORITHM'S  
: ABILITY TO SET BOTH N AND Z ON A -0 RESULT.  
: NOTE THAT -0 IS AN UNDERFLOW CONDITION.  
:  
HH11:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
MOV #42200,R4  
LDFPS R4 ; SET FID AND FIU BITS.  
MOV #HH12,@#$TMP2  
MOV #HHP8,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV #HHP9,R0 ;: SRC OPERAND.  
HH12: ADDD (R0),ACO ;: TEST INSTRUCTION  
STFPS R5 ;: GET FPS  
MOV #HHSUM,R0  
STD ACO,(R0) ;: STORE ACO  
BIS #100014,R4 ; SET EXP FPS.  
CMP R4,R5 ; FPS RIGHT ??  
BNE 1$ ; NO.  
JSR R5,CHECK4 ; YES, DATA RIGHT ??  
HHP7,HHSUM  
BEQ HH17 ; BR IF SO.  
  
1$: MOV R5,$TMP3  
MOV R4,$TMP4  
MOV #HHP9,$TMP5  
MOV #HHP8,$TMP6  
MOV #HHP7,$TMP7  
MOV #HHSUM,$TMP10  
ERROR 160  
  
: TEST THAT CC BITS ARE CLEARED BY R/T  
:  
HH17:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
MOV #200,R4  
LDFPS R4  
MOV #HH18,@#$TMP2  
MOV #HHP8,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV #HHP8,R0 ;: SRC TOO.  
HH18: ADDD (R0),ACO ;: TEST INSTRUCTION  
STFPS R5 ;: GET FPS  
MOV #HHSUM,R0
```

(1)	025234	174010				STD	ACO,(R0)	:: STORE ACO
3175	025236	020405				CMP	R4,R5	: FPS RIGHT ??
3176	025240	001005				BNE	1\$: NO.
3177	025242	004537	004312			JSR	R5,CHECK4	: YES, DATA RIGHT ??
3178	025246	025576	025446			HHP10,HHSUM		
3179	025252	001421				BEQ	HH21	: YES.
3180								
3181	025254	010537	001166		1\$:	MOV	R5,\$TMP3	
3182	025260	010437	001170			MOV	R4,\$TMP4	
3183	025264	012737	025556	001172		MOV	#HHP8,\$TMP5	
3184	025272	013737	001172	001174		MOV	\$TMP5,\$TMP6	
3185	025300	012737	025576	001176		MOV	#HHP10,\$TMP7	
3186	025306	012737	025446	001200		MOV	#HHSUM,\$TMP10	
3187	025314	104160				ERROR	160	
3188								
3189								:: TEST THAT N BIT IS SET BY R/T
3190								
3191	025316					HH21:		
(1)	025316	104411				LUPERR		:: LOOP HERE ON ERROR IF SWR9 = 1.
3192	025320	012704	000200			MOV	#200,R4	
3193	025324	170104				LDFPS	R4	
3194	025326	012737	025346	001164		MOV	#HH22,@#\$TMP2	
3195	025334	012700	025526			MOV	#HHP5,R0	
(1)	025340	172410				LDD	(R0),ACO	:: LOAD ACO
3196	025342	012700	025526			MOV	#HHP5,R0	: SRC TOO.
3197	025346	172010			HH22:	ADDD	(R0),ACO	: TEST INSTRUCTION
3198	025350	170205				STFPS	R5	: GET FPS
3199	025352	012700	025446			MOV	#HHSUM,R0	
(1)	025356	174010				STD	ACO,(R0)	:: STORE ACO
3200	025360	052704	000010			BIS	#10,R4	: SET EXP FPS.
3201	025364	020405				CMP	R4,R5	: FPS RIGHT ??
3202	025366	001005				BNE	1\$: NO.
3203	025370	004537	004312			JSR	R5,CHECK4	: YES, DATA RIGHT ??
3204	025374	025606	025446			HHP11,HHSUM		
3205	025400	001421				BEQ	2\$: YES.
3206								
3207	025402	010537	001166		1\$:	MOV	R5,\$TMP3	
3208	025406	010437	001170			MOV	R4,\$TMP4	
3209	025412	012737	025526	001172		MOV	#HHP5,\$TMP5	
3210	025420	013737	001172	00117		MOV	\$TMP5,\$TMP6	
3211	025426	012737	025606	001176		MOV	#HHP11,\$TMP7	
3212	025434	012737	025446	001200		MOV	#HHSUM,\$TMP10	
3213	025442	104160				ERROR	160	
3214	025444	000464			2\$:	BR	HHDONE	
3215								
3216								
3217	025446	000000	000000	000000	HHSUM:	.FLT4	0	
3218	025456	000452	125252	125252	HHP0:	000452,125252,125252,125253		
3219	025466	000252	125252	125252	HHP1:	000252,125252,125252,125252		
3220	025476	000600	000000	000000	HHP2:	000600,0,0,0	: HHP0 + HHP1 NORMALIZED.	
3221	025506	000400	000000	000000	HHP3:	000400,0,0,0	: HHP0 + HHP1 WITH...	
3222							: ...BAD NORMALIZATION	
3223	025516	000000	000000	000000	HHP4:	0,0,0,0		
3224	025526	100200	000000	000000	HHP5:	100200,0,0,0		
3225	025536	000300	000000	000000	HHP6:	000300,0,0,0		
3226	025546	100000	000000	000000	HHP7:	100000,0,0,0	: HHP7 - HHP8 + HHP9 (NEG ZERO)	

```
3227                                     ;      - HHP5 + HHP6
3228 J25556 000200 000000 000000 HHP8: 000200,0,0,0
3229 025566 100300 000000 000000 HHP9: 100300,0,0,0
3230 025576 000400 000000 000000 HHP10: 000400,0,0,0      ;HHP10 = HHP8 + HHP8
3231 025606 100400 000000 000000 HHP11: 100400,0,0,0      ;HHP11 = HHP5 + HHP5
3232
3233 025616                                HHDONE:
  (1) 025616 104412                        CLRFP5      ;; CLEAR FP STATUS...
  (3) 025620 000400                        BR          TST40      ;:...AND PROCEED.
```


3245
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 025622 000004
3246
3247
3248
3249 025624 012737 026526 000244
3250 025632 104411
3251 025634 012704 000200
3252 025640 170104
3253 025642 052704 000006
3254 025646 010437 001170
3255 025652 012737 026560 001172
3256 025660 012737 026570 001174
3257 025666 012737 026550 001176
3258 025674 012737 026540 001200
3259 025702 012737 025726 001164
3260 025710 005037 026536
3261 025714 012700 026570
(1) 025720 172410
3262 025722 012700 026560
3263 025726 172010
3264 025730 000240
3265 025732 170205
3266 025734 010537 001166
3267 025740 012700 026540
(1) 025744 174010
3268 025746 005737 026536
3269 025752 001013
3270
3271 025754 023737 001166 001170
3272 025762 001005
3273 025764 004537 004312
3274 025770 026550 026540
3275 025774 001403
3276 025776 104162
3277 026000 000401
3278 026002 104163
3279
3280
3281
3282 026004
(1) 026004 104411
3283 026006 012704 001200
3284 026012 170104
3285 026014 052704 100006
3286 026020 010437 001170

```
.....  
*TEST 40 TEST OVERFLOW/UNDERFLOW USING ADDD  
*  
* TEST THAT THE OVERFLOW/UNDERFLOW ALGORITHM CAN CORRECTLY  
* DETECT THOSE CONDITIONS AND TAKE THE APPROPRIATE ACTION.  
* EACH CONDITION IS TESTED TWICE, ONCE WITH TRAPS  
* ENABLED (FIU AND FIV = 1), AND AGAIN WITH TRAPS  
* DISABLED (FIU AND FIV = 0).  
* WHEN TRAPS ARE ENABLED, THE RETURNED FEC IS ALSO TESTED.  
*  
.....  
TST40: SCOPE  
:  
: OVERFLOW WITH TRAP DISABLED (FIV = 0)  
:  
GG1: MOV #GG244,FPVEC ; SET OUR TRAP CATCHER.  
LUPERR ; LOOP HERE ON ERROR IF SWR9 = 1.  
MOV #200,R4 ; FIV = 0.  
LDFPS R4  
BIS #6,R4  
MOV R4,$TMP4 ; EXP FPS.  
MOV #GGP1,$TMP5 ; OPERANDS FOR OUTPUT.  
MOV #GGP2,$TMP6  
MOV #GGP0,$TMP7 ; EXP SUM  
MOV #GGSUM,$TMP10 ; RECD SUM  
MOV #1$,@,$TMP2  
CLR GGFEC ; USE AS TRAP FLAG.  
MOV #GGP2,R0  
LDD (R0),ACO ; LOAD ACO  
MOV #GGP1,R0 ; SRC TOO.  
1$: ADDD (R0),ACO ; TEST INSTRUCTION...  
NOP ; ...SHOULDN'T TRAP.  
STFPS R5 ; GET FPS  
MOV R5,$TMP3  
MOV #GGSUM,R0  
STD ACO,(R0) ; STORE ACO  
TST GGFEC ; DID TRAP OCCUR ??  
BNE 4$ ; YES, BAD NEWS !!  
2$: CMP $TMP3,$TMP4 ; NO, FPS RIGHT ??  
BNE 3$ ; NO.  
JSR R5,CHECK4 ; YES, SUM = OVERFLOW ??  
GGP0,GGSUM  
BEQ GG2 ; YES, PROCEED.  
3$: ERROR 162 ; OVER SUM OR FPS BAD.  
SKP1  
4$: ERROR 163 ; TRAPPED WITH FIV = 0.  
:  
: SAME OPERANDS, TRAP ENABLED (FIV = 1).  
:  
GG2: LUPERR ; LOOP HERE ON ERROR IF SWR9 = 1.  
MOV #1200,R4 ; SET FIV.  
LDFPS R4  
BIS #100006,R4  
MOV R4,$TMP4 ; EXP FPS.
```

3287	026024	012737	026620	001176		MOV	#GGP6,\$TMP7	:	EXP SUM.
3288	026032	012737	026056	001164		MOV	#1\$,@#\$TMP2	:	
3289	026040	005037	026536			CLR	GGFEC	:	CLEAR TRAP FLAG.
3290	026044	012700	026570			MOV	#GGP2,R0	:	
(1)	026050	172410				LDD	(R0),ACO	::	LOAD ACO
3291	026052	012700	026560			MOV	#GGP1,R0	:	SRC TOO.
3292	026056	172010			1\$:	ADDD	(R0),ACO	:	TEST INSTRUCTION...
3293	026060	000240				NOP		:	...SHOULD TRAP.
3294	026062	170205				STFPS	R5	:	GET FPS
3295	026064	010537	001166			MOV	R5,\$TMP3	:	
3296	026070	012700	026540			MOV	#GGSUM,R0	:	
(1)	026074	174010				STD	ACO,(R0)	::	STORE ACO
3297	026076	005737	026536			TST	GGFEC	:	DID IT TRAP ??
3298	026102	001432				BEQ	4\$:	NO, BAD NEWS !!
3299								:	
3300	026104	004537	004312			JSR	R5,CHECK4	:	YES, IS SUM RIGHT ??
3301	026110	026620	026540			GGP6,GGSUM		:	
3302	026114	001401				BEQ	2\$:	BR IF SO.
3303	026116	104162				ERROR	162	:	SUM WRONG.
3304	026120	010437	001166		2\$:	MOV	R4,\$TMP3	:	MOVE EXP FPS.
3305	026124	012737	000010	001170		MOV	#10,\$TMP4	:	SET EXP' FEC.
3306	026132	170205				STFPS	R5	:	GET FPS AGAIN
3307	026134	020537	001166			CMP	R5,\$TMP3	:	IS IT RIGHT ??
3308	026140	001004				BNE	3\$:	NO.
3309	026142	023737	026536	001170		CMP	GGFEC,\$TMP4	:	FEC RIGHT (OVFLO) ??
3310	026150	001410				BEQ	GG3	:	YES.
3311	026152	010537	001172		3\$:	MOV	R5,\$TMP5	:	
3312	026156	013737	026536	001174		MOV	GGFEC,\$TMP6	:	
3313	026164	104165				ERROR	165	:	FPS/FEC WRONG ON OVERFLOW.
3314	026166	000401				SKP1		:	
3315	026170	104164			4\$:	ERROR	164	:	DIDN'T TRAP FIV - 1.
3316								:	
3317								:	
3318								:	
3319								:	
3320	026172							:	
(1)	026172	104411			GG3:	LUPERR		::	LOOP HERE ON ERROR IF SWR9 1.
3321	026174	012704	000200			MOV	#200,R4	:	FIU - 0
3322	026200	170104				LDFPS	R4	:	
3323	026202	052704	000004			BIS	#4,R4	:	
3324	026206	010437	001170			MOV	R4,\$TMP4	:	EXP FPS.
3325	026212	012737	026600	001172		MOV	#GGP3,\$TMP5	:	
3326	026220	012737	026610	001174		MOV	#GGP4,\$TMP6	:	
3327	026226	012737	026550	001176		MOV	#GGP0,\$TMP7	:	
3328	026234	012737	026260	001164		MOV	#1\$,@#\$TMP2	:	
3329	026242	005037	026536			CLR	GGFEC	:	CLEAR TRAP FLAG.
3330	026246	012700	026610			MOV	#GGP4,R0	:	
(1)	026252	172410				LDD	(R0),ACO	::	LOAD ACO
3331	026254	012700	026600			MOV	#GGP3,R0	:	SRC OPERAND.
3332	026260	172010			1\$:	ADDD	(R0),ACO	:	TEST INSTRUCTION...
3333	026262	000240				NOP		:	...SHOULDN'T TRAP.
3334	026264	170205				STFPS	R5	:	GET FPS
3335	026266	010537	001166			MOV	R5,\$TMP3	:	
3336	026272	012700	026540			MOV	#GGSUM,R0	:	
(1)	026276	174010				STD	ACO,(R0)	::	STORE ACO
3337	026300	005737	026536			TST	GGFEC	:	DID IT TRAP ??

3391	026536	000000			GGFEC:	0		; SAVED FEC CODE ON TRAP.
3392	026540	000000	000000	000000	GGSUM:	.FLT4	0	
3393	026550	000000	000000	000000	GGP0:	0,0,0,0		; RESULT WHEN FIU/FIV = 0.
3394	026560	077600	000000	000000	GGP1:	77600,0,0,0		
3395	026570	077601	000001	000001	GGP2:	77601,1,1,1		; 1 + 2 = OVERFLOW.
3396	026600	100200	000000	000000	GGP3:	100200,0,0,0		
3397	026610	000201	000001	000001	GGP4:	000201,1,1,1		; 3 + 4 = UNDERFLOW.
3398	026620	000000	100000	100000	GGP6:	0,100000,100000,100001		; SUM OF 1 + 2 (FIV = 1).
3399	026630	076400	000200	000200	GGP7:	76400,200,200,200		; SUM OF 3 + 4 (FIU = 1)
3400								
3401	026640				GGDONE:			
(1)	026640	104412			CLRFPS			:: CLEAR FP STATUS...
(3)	026642	000400			BR	TST41		::...AND PROCEED.

3408
(3)
(4)
(4)
(4)
(3)
(2) 026644 000004
3409
3410
3411
3412 026646
(1) 026646 104411
3413 026650 012704 000200
3414 026654 170104
3415 026656 012737 026716 001164
3416 026664 012737 042106 055170
3417 026672 013737 055170 055236
3418 026700 012700 027526
(1) 026704 172410
3419 026706 012700 027536
3420 026712 010037 001166
3421 026716 177420
3422 026720 020027 027542
3423 026724 001407
3424 026726 010037 001170
3425 026732 012737 027542 001172
3426 026740 104172
3427 026742 000425
3428 026744 170205
3429 026746 010537 001166
3430 026752 012700 027516
(1) 026756 174010
3431 026760 004537 004312
3432 026764 027606 027516
3433 026770 001412
3434 026772 012737 027536 001170
3435 027000 012737 027606 001172
3436 027006 012737 027516 001174
3437 027014 104173
3438
3439
3440
3441 027016
(1) 027016 104411
3442 027020 012704 000200
3443 027024 170104
3444 027026 012737 027070 001164
3445 027034 012737 043104 055170
3446 027042 013737 055170 055236
3447 027050 012700 027526
(1) 027054 172410
3448 027056 170001
3449 027060 012700 027536
3450 027064 010037 001166
3451 027070 177420
3452 027072 020027 027546

```
*****
*TEST 41      LDCFD AND LDCDF -- FRSC MODE 2
*
*THIS IS A TEST OF LDCFD AND LDCDF.
*
*****
TST41: SCOPE
:TEST FOR CORRECT AUTO INCREMENT CONSTANT.
JJ1:
      LUPERR                                ;; LOOP HERE ON ERROR IF SWR9 = 1.
      MOV      #200,R4
      LDFPS   R4
      MOV      #JJ2,@#STMP2
      MOV      #'FD,EM172X
      MOV      EM172X,EM173X
      MOV      #JJP1,R0
      LDD     (R0),ACO                        ;; LOAD ACO
      MOV      #JJP2,R0
      MOV      R0,$TMP3                       ; RO BEFORE.
      LDCFD   (R0)+,ACO                       ; CONVERT P2
      CMP     R0,#JJP2+4                       ; IS RO CORRECT
      BEQ     1$                               ; YES.
      MOV      R0,$TMP4
      MOV      #JJP2+4,$TMP5
      ERROR   172
      BR      JJ8
1$:    STFPS  R5                               ;GET FPS
      MOV      R5,$TMP3
      MOV      #JJDAT,R0
      STD     ACO,(R0)                        ;; STORE ACO
      JSR     R5,CHECK4                       ; DATA RIGHT ??
      JJP7,JJDAT
      BEQ     JJ8                               ; YES.
      MOV      #JJP2,$TMP4                   ; SRC DATA
      MOV      #JJP7,$TMP5                   ; EXP
      MOV      #JJDAT,$TMP6
      ERROR   173
:
: NOW AGAIN D TO F.
JJ8:
      LUPERR                                ;; LOOP HERE ON ERROR IF SWR9 = 1.
      MOV      #200,R4
      LDFPS   R4
      MOV      #JJ9,@#STMP2
      MOV      #'DF,EM172X
      MOV      EM172X,EM173X
      MOV      #JJP1,R0
      LDD     (R0),ACO                        ;; LOAD ACO
      SETF
      MOV      #JJP2,R0
      MOV      R0,$TMP3                       ; RO BEFORE.
      LDCDF   (R0)+,ACO                       ; CONVERT P2.
      CMP     R0,#JJP2+10                     ;RO RIGHT ??
```

3453	027076	001407			BEQ	1\$: YES.
3454	027100	010037	001170		MOV	R0,\$TMP4		: NO
3455	027104	012737	027546	001172	MOV	#JJP2+10,\$TMP5		
3456	027112	104172			ERROR	172		
3457	027114	000426			BR	JJ17		
3458	027116	170205			STFPS	R5	1\$:	
3459	027120	010537	001166		MOV	R5,\$TMP3		
3460	027124	170011			SETD			
3461	027126	012700	027516		MOV	#JJDAT,R0		
(1)	027132	174010			STD	AC0,(R0)		:: STORE AC0
3462	027134	004537	004312		JSR	R5,CHECK4		: DATA RIGHT ??
3463	027140	027616	027516		JJP8,JJDAT			
3464	027144	001412			BEQ	JJ17		: YES.
3465	027146	012737	027536	001170	MOV	#JJP2,\$TMP4		: SRC DATA
3466	027154	012737	027616	001172	MOV	#JJP8,\$TMP5		: EXP
3467	027162	012737	027516	001174	MOV	#JJDAT,\$TMP6		: RECD.
3468	027170	104173			ERROR	173		
3469								
3470								
3471								
3472	027172							
(1)	027172	104411			JJ17:			
3473	027174	012704	000200		LUPERR			:: LOOP HERE ON ERROR IF SWR9 1.
3474	027200	170104			MOV	#200,R4		
3475	027202	012737	027230	001164	LDFPS	R4		
3476	027210	012737	042106	055236	MOV	#JJ18,@#\$TMP2		
3477	027216	012700	027526		MOV	#FD,EM173X		
(1)	027222	172410			MOV	#JJP1,R0		
3478	027224	012700	027536		LDD	(R0),AC0		:: LOAD AC0
3479	027230	177420			MOV	#JJP2,R0		
3480	027232	170205			JJ18:	LDCFD (R0)+,AC0		: CONVERT P2.
3481	027234	010537	001166		STFPS	R5		
3482	027240	012700	027516		MOV	R5,\$TMP3		
(1)	027244	174010			MOV	#JJDAT,R0		
3483	027246	004537	004312		STD	AC0,(R0)		:: STORE AC0
3484	027252	027606	027516		JSR	R5,CHECK4		: IS IT RIGHT ??
3485	027256	001412			JJP7,JJDAT			
3486	027260	012737	027536	001170	BEQ	JJ21		: YES.
3487	027266	012737	027606	001172	MOV	#JJP2,\$TMP4		
3488	027274	012737	027516	001176	MOV	#JJP7,\$TMP5		
3489	027302	104173			MOV	#JJDAT,\$TMP7		
3490					ERROR	173		
3491								
3492								
3493	027304							
(1)	027304	104411			JJ21:			
3494	027306	012704	000200		LUPERR			:: LOOP HERE ON ERROR IF SWR9 1.
3495	027312	170104			MOV	#200,R4		
3496	027314	012737	027334	001164	LDFPS	R4		
3497	027322	012700	027576		MOV	#JJ22,@#\$TMP2		
(1)	027326	172410			MOV	#JJP6,R0		
3498	027330	012700	027556		LDD	(R0),AC0		:: LOAD AC0
3499	027334	177420			MOV	#JJP4,R0		
3500	027336	170205			JJ22:	LDCFD (R0)+,AC0		: CONVERT P4.
3501	027340	010537	001166		STFPS	R5		
3502	027344	012700	027516		MOV	R5,\$TMP3		
					MOV	#JJDAT,R0		

```

(1) 027350 174010 STD ACO,(R0) ;; STORE ACO
3503 027352 004537 004312 JSR R5,CHECK4 ; IS IT RIGHT ??
3504 027356 027566 027516 JJP5,JJDAT
3505 027362 001412 BEQ JJ27 ; YES.
3506 027364 012737 027556 001170 MOV #JJP4,$TMP4
3507 027372 012737 027566 001172 MOV #JJP5,$TMP5
3508 027400 012737 027516 001174 MOV #JJDAT,$TMP6
3509 027406 104173 ERROR 173
3510
3511 ; NOW LOAD AND CONVERT 0.
3512
3513 JJ27:
(1) 027410 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.
3514 027412 104411 MOV #200,R4
3515 027416 170104 LDFPS R4
3516 027420 012737 027440 001164 MOV #JJ28,$TMP2
3517 027426 012700 027576 MOV #JJP6,R0
(1) 027432 172410 LDD (R0),ACO ;; LOAD ACO
3518 027434 012700 027526 MOV #JJP1,R0
3519 027440 177420 JJ28: LDCFD (R0)+,ACO ; CONVERT P1.
3520 027442 170205 STFPS R5
3521 027444 010537 001166 MOV R5,$TMP3
3522 027450 012700 027516 MOV #JJDAT,R0
(1) 027454 174010 STD ACO,(R0) ;; STORE ACO
3523 027456 004537 004312 JSR R5,CHECK4 ; GOT 0 ??
3524 027462 027526 027516 JJP1,JJDAT
3525 027466 001412 BEQ JJ29 ; YES.
3526 027470 012737 027526 001170 MOV #JJP1,$TMP4
3527 027476 012737 027526 001172 MOV #JJP1,$TMP5
3528 027504 012737 027516 001174 MOV #JJDAT,$TMP6
3529 027512 104173 ERROR 173
3530
3531 027514 000444 JJ29: BR JJDONE
3532
3533 027516 000000 000000 000000 JJDAT: 0,0,0,0
3534 027526 000000 000000 000000 JJP1: 0,0,0,0
3535 027536 000577 177776 177777 JJP2: 000577,177776,177777,177776
3536 027546 005201 000000 000000 JJP3: 005201,0,0,0
3537 027556 100577 177776 177777 JJP4: 100577,177776,177777,177776
3538 027566 100577 177776 000000 JJP5: 100577,177776,0,0
3539 027576 000252 125252 125252 JJP6: 000252,125252,125252,125252
3540 027606 000577 177776 000000 JJP7: 000577,177776,0,0
3541 027616 000577 177777 000000 JJP8: 000577,177777,0,0
3542
3543 JJDONE:
(1) 027626 CLRFPs ;; CLEAR FP STATUS...
(3) 027630 000400 BR TST42 ;...AND PROCEED.

```

3555
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 027632 000004
3556
3557
3558
3559 027634 004737 030326
3560 027640 000000 000000 000000
3561 027650 000000 000000 000000
3562 027660 000204
3563 027662 036440
3564
3565
3566
3567 027664 004737 030326
3568 027670 000000 000000 000000
3569 027700 025252 052525 125252
3570 027710 000200
3571 027712 037040
3572
3573
3574
3575 027714 004737 030326
3576 027720 000000 000000 000000
3577 027730 125252 125252 052525
3578 027740 000210
3579 027742 036040
3580
3581
3582
3583 027744 004737 030326
3584 027750 025252 052525 125252
3585 027760 000000 000000 000000
3586 027770 000210
3587 027772 036040
3588
3589
3590
3591 027774 004737 030326
3592 030000 125252 125252 052525
3593 030010 000000 000000 000000
3594 030020 000200
3595 030022 037040
3596
3597
3598
3599 030024 004737 030326

```
.....  
*TEST 42      CMPD -- FSRC MODE 1  
*  
*TEST THE CMPD INSTRUCTION WITH A VARIETY OF OPERANDS.  
*A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE  
*INSTRUCTION, AND CHECK THE RESULTS.  
*NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE  
*OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE  
*OFFENDING INSTRUCTION ITSELF.  
*  
.....  
TST42: SCOPE  
*  
*TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)  
*  
KK1:  JSR      PC,KKSUB  
      0,0,0,0      :AC  
      0,0,0,0      :FSRC  
      204          : EXP FPS AFTER EXECUTION.  
      .ASCII ' -'  : RELATIONAL OPERATOR FOR ERROR.  
*  
*TEST CMPD WITH (AC=0) AND FSRC POSITIVE.  
*  
KK2:  JSR      PC,KKSUB  
      0,0,0,0      :AC  
      25252,52525,125252,52525 :FSRC  
      200          : EXP FPS.  
      .ASCII ' >'  : FSRC > AC  
*  
*TEST CMPD WITH (AC=0) AND FSRC NEGATIVE  
*  
KK3:  JSR      PC,KKSUB  
      0,0,0,0      :AC  
      125252,125252,52525,125252 :FSRC  
      210          : EXP FPS.  
      .ASCII ' <'  : FSRC < AC  
*  
*TEST CMPD WITH (FSRC=0) AND AC POSITIVE  
*  
KK4:  JSR      PC,KKSUB  
      25252,52525,125252,52525 :AC  
      0,0,0,0      :FSRC  
      210          : EXP FPS.  
      .ASCII ' <'  : FSRC < AC  
*  
*TEST CMPD WITH (FSRC=0) AND AC NEGATIVE  
*  
KK5:  JSR      PC,KKSUB  
      125252,125252,52525,125252 :AC  
      0,0,0,0      :FSRC  
      200          : EXP FPS.  
      .ASCII ' >'  : FSRC > AC  
*  
*TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE  
*  
KK6:  JSR      PC,KKSUB
```



```
3600 030030 052525 125252 052525 52525,125252,52525,125252 ; AC
3601 030040 125252 052525 125252 125252,52525,125252,52525 ; FSRC
3602 030050 000210 210 ; EXP FPS.
3603 030052 036040 .ASCII ' <' ; FSRC < AC
3604
3605 :TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
3606
3607 030054 004737 030326 KK7: JSR PC,KKSUB
3608 030060 125252 052525 125252 125252,52525,125252,52525 ; AC
3609 030070 052525 125252 052525 52525,125252,52525,125252 ; FSRC
3610 030100 000200 200 ; EXP FPS.
3611 030102 037040 .ASCII ' >' ; FSRC > AC
3612
3613 :TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
3614 :AND EAC LESS THAN EFSRC.
3615
3616 030104 004737 030326 KK8: JSR PC,KKSUB
3617 030110 012345 067654 032101 12345,67654,32101,23456 ; AC
3618 030120 023456 076543 021012 23456,76543,21012,34567 ; FSRC
3619 030130 000200 200 ; EXP FPS.
3620 030132 037040 .ASCII ' >' ; FSRC > AC
3621
3622 :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND
3623 :EAC GREATER THAN EFSRC.
3624
3625 030134 004737 030326 KK9: JSR PC,KKSUB
3626 030140 045676 054321 012345 45676,54321,12345,67654 ; AC
3627 030150 034567 065432 101234 34567,65432,101234,56765 ; FSRC
3628 030160 000210 210 ; EXP FPS.
3629 030162 036040 .ASCII ' <' ; FSRC < AC
3630
3631 :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
3632
3633 030164 004737 030326 KK10: JSR PC,KKSUB
3634 030170 012345 067012 034567 12345,67012,34567,012345 ; AC
3635 030200 012345 067012 034567 12345,67012,34567,012345 ; FSRC
3636 030210 000204 204 ; EXP FPS.
3637 030212 036440 .ASCII ' =' ; FSRC = AC
3638
3639 :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
3640 :AND FSRC GREATER THAN AC.
3641
3642 030214 004737 030326 KK11: JSR PC,KKSUB
3643 030220 012345 067012 034567 12345,67012,34567,012345 ; AC
3644 030230 012345 070123 045670 12345,70123,45670,123456 ; FSRC
3645 030240 000200 200 ; EXP FPS.
3646 030242 037040 .ASCII ' >' ; FSRC > AC
3647
3648 :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
3649 :AND AC GREATER THAN FSRC.
3650
3651 030244 004737 030326 KK12: JSR PC,KKSUB
3652 030250 054321 076543 021076 54321,76543,21076,54321 ; AC
3653 030260 054321 065432 107654 54321,65432,107654,32107 ; FSRC
3654 030270 000210 210 ; EXP FPS.
3655 030272 036040 .ASCII ' <' ; FSRC < AC
```

3656
 3657
 3658
 3659
 3660 030274 004737 030326
 3661 030300 112345 043210 076543
 3662 030310 112345 054321 007654
 3663 030320 000210
 3664 030322 036040
 3665
 3666 030324 000470
 3667
 3668
 3669
 3670
 3671
 3672
 3673
 3674
 3675
 3676
 3677
 3678
 3679
 3680
 3681
 3682
 3683
 3684
 3685 030326 012637 001164
 3686 030332 104411
 3687 030334 013701 001164
 3688 030340 012700 000217
 3689 030344 170100
 3690 030346 010100
 3691 030350 172410
 3692 030352 010037 001174
 3693 030356 062700 000010
 3694 030362 010037 001172
 3695 030366 000240
 3696 030370 173410
 3697 030372 000240
 3698 030374 170205
 3699 030376 010537 001166
 3700 030402 016137 000020 001170
 3701 030410 012700 030476
 (1) 030414 174010
 3702 030416 010137 030426
 3703 030422 004537 004312
 3704 030426 000000 030476
 3705 030432 001407
 3706 030434 010137 001172
 3707 030440 012737 030476 001174
 3708 030446 104202
 3709 030450 000410
 3710

TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC,
 AND AC GREATER THAN FSRC

KK13: JSR PC,KKSUB
 112345,43210,76543,21076 ; AC
 112345,54321,07654,32107 ; FSRC
 210 ; EXP FPS.
 .ASCII ' <' ; FSRC < AC
 BR KKDONE ; FINALLY !.

SUBROUTINE TO SET-UP, EXECUTE AND CHECK THE CMPD INSTRUCTION.

CALL:
 JSR PC,KKSUB ; AC OPERAND
 .WORD X,X,X,X ; FSRC OPERAND
 .WORD X,X,X,X ; EXPECTED FPS.
 .WORD X ; RELATIONAL OPERATOR...
 .ASCII ' =' ; ...FOR ERROR MESSAGE.
 ; RETURN TO CALLER+24

OPERANDS ARE SELECTED FROM THE CALLING PC.
 THE FPS IS INITIALIZED TO DOUBLE MODE WITH ALL CONDITION CODE
 BITS SET (000217). THEN THE COMPARE INSTRUCTION IS EXECUTED.
 AFTER EXECUTION, THE AC OPERAND SHOULD BE UNCHANGED, AND THE
 FPS SHOULD EQUAL THE PREDICTED VALUE. IF SO RETURN.
 OTHERWISE, REPORT THE APPROPRIATE ERROR, AND RETURN.

KKSUB: MOV (SP)+,\$TMP2 ; SAVE CALL PC AS ERROR PC...
 LUPERR ; LOOP HERE ON ERROR IF SWR9 1.
 MOV \$TMP2,R1 ; ...AND USE AS ARG POINTER.
 MOV #217,R0
 LDFPS R0 ; INITIALIZE FPS.
 MOV R1,R0 ; LOAD UP...
 LDD (R0),AC0 ; ...AC OPERAND
 MOV R0,\$TMP6
 ADD #10,R0 ; POINT TO FSRC OPERAND.
 MOV R0,\$TMP5
 NOP ; FOR SCOPING.
 2\$: CMPD (R0),AC0 ; EXECUTE THE TEST INSTRUCTION.
 NOP
 STFPS R5 ; GET FPS AFTER INSTRUCTION.
 MOV R5,\$TMP3
 MOV 20(R1),\$TMP4 ; SET EXPECTED FPS.
 MOV #10\$,R0
 STD AC0,(R0) ; STORE AC0
 MOV R1,\$\$; ORIGINAL AC ADDRESS.
 JSR R5,CHECK4
 3\$: O,10\$; EXP VS RECD.
 BEQ 4\$; BR IF AC IS STILL OK.
 MOV R1,\$TMP5 ; AC BEFORE.
 MOV #10\$, \$TMP6 ; AC AFTER.
 ERROR 202 ; REPORT AC CHANGED.
 BR 5\$

```
3711 030452 023737 001166 001170 4$: CMP $TMP3,$TMP4 ; FPS RIGHT ?  
3712 030460 001404 BEQ 5$ ; BR IF SO.  
3713 030462 016137 000022 055332 MOV 22(R1),EM201X ; SET CORRECT OPERATER IN ERR.  
3714 030470 104201 ERROR 201 ; REPORT FPS WRONG.  
3715  
3716 030472 000161 000024 5$: JMP 24(R1) ; RETURN.  
3717  
3718 030476 000000 000000 000000 10$: .WORD 0,0,0,0 ; TEMP DATA BUFFER.  
3719  
3720 030506 KKDONE:  
(1) 030506 104412 CLRFP  
(3) 030510 000400 BR TST43 ;: CLEAR FP STATUS...  
;:...AND PROCEED.
```

3729
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 030512 000004
3730
3731
3732
3733 030514 012737 001172 001166
3734 030522 012737 001176 001170
3735 030530 104411
3736 030532 012704 040200
3737 030536 170104
3738 030540 012737 030566 001164
3739 030546 012737 030640 000244
3740 030554 012700 031144
(1) 030560 172410
3741 030562 012700 031144
3742 030566 174410
3743 030570 000240
3744 030572 012737 140204 001172
3745 030600 012737 000004 001174
3746 030606 170205
3747 030610 010537 001176
3748 030614 170304
3749 030616 010437 001200
3750 030622 004537 004302
3751 030626 001172 001176
3752 030632 001405
3753
3754 030634 104203
3755 030636 000403
3756
3757 030640 022626
3758 030642 004737 031116
3759
3760
3761
3762 030646
(1) 030646 104411
3763 030650 012704 040200
3764 030654 170104
3765 030656 012737 030704 001164
3766 030664 012737 030756 000244
3767 030672 012700 031154
(1) 030676 172410
3768 030700 012700 031144
3769 030704 174410
3770 030706 000240
3771 030710 012737 140200 001172
3772 030716 012737 000004 001174
3773 030724 170205

```
*****  
*TEST 43 DIVD -- FSRC MODE 1, WITH DIVISOR - 0  
*  
*THIS IS A TEST OF THE DIVD INSTRUCTION WITH A  
*ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH  
*TRAP ENABLED AND TRAPS DISABLED.  
*  
*****  
TST43: SCOPE  
:  
: FIRST BOTH DIVISOR AND DIVIDEND = 0, WITH TRAP DISABLED.  
LL0: MOV #STMP5,STMP3 ; EXPD STATS POINTER.  
MOV #STMP7,STMP4 ; RECD STATS POINTER.  
LUPERR ; LOOP HERE ON ERROR IF SWR9 - 1.  
MOV #40200,R4 ; SET UP FPS...  
LDFPS R4 ; ...WITH FID = 1.  
MOV #20$,STMP2  
MOV #2$,FPVEC ; SET TRAP CATCHER.  
MOV #LLP1,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV #LLP1,R0 ;: DIVISOR = 0  
20$: DIVD (R0),ACO ;: 0/0 = ERROR.  
240  
MOV #140204,STMP5 ; EXPECTED FPS...  
MOV #4,STMP6 ; ...AND FEC.  
STFPS R5  
MOV R5,STMP7 ; RECEIVED FPS...  
STST R4  
MOV R4,STMP10 ; ...AND FEC.  
JSR R5,CHECK2  
STMP5,STMP7  
BEQ LL2 ; BR IF BOTH ARE RIGHT.  
ERROR 203 ; FPS OR FEC WRONG.  
BR LL2  
2$: CMP (SP)+,(SP)+ ; IT TRAPPED ?????  
JSR PC,LL244 ; REPORT ERROR AND RETURN.  
:  
: NOW DIVIDE +N/0 WITH TRAP DISABLED.  
LL2:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
MOV #40200,R4  
LDFPS R4  
MOV #20$,STMP2  
MOV #2$,FPVEC  
MOV #LLP2,R0  
LDD (R0),ACO ;: LOAD ACO  
MOV #LLP1,R0 ;: DIVISOR = 0  
20$: DIVD (R0),ACO ;: +N/0 = ERROR.  
240  
MOV #140200,STMP5 ; EXPECTED FPS...  
MOV #4,STMP6 ; ...AND FEC.  
STFPS R5
```

```

3774 030726 010537 001176      MOV    R5,$TMP7      ; RECEIVED FPS...
3775 030732 170305              STST   R5
3776 030734 010537 001200      MOV    R5,$TMP10     ; ...AND FEC.
3777 030740 004537 004302      JSR    R5,CHECK2
3778 030744 001172 001176      $TMP5,$TMP7
3779 030750 001405              BEQ    LL4           ; BR IF BOTH OK.
3780
3781 030752 104203              ERROR  203          ; FPS OR FEC WRONG.
3782 030754 000403              BR     LL4
3783
3784 030756 022626 031116      2$:    CMP    (SP)+,(SP)+ ; IT TRAPPED, FIX STACK.
3785 030760 004737              JSR    PC,LL244     ; REPORT AND RETURN.
3786
3787      ; FINALLY, +N/O WITH TRAP ENABLED.
3788
3789      LL4:
(1) 030764 104411              LUPERR              ;; LOOP HERE ON ERROR IF SWR9 1.
3790 030766 012704 000200      MOV    #200,R4      ; SET UP FPS. TRAP ENABLED.
3791 030772 170104              LDFPS  R4
3792 030774 012737 031022 001164  MOV    #20,$TMP2
3793 031002 012737 031046 000244  MOV    #1$,FPVEC    ; SET TRAP VECTOR.
3794 031010 012700 031154      MOV    #LLP2,R0
(1) 031014 172410              LDD    (R0),ACO     ;; LOAD ACO
3795 031016 012700 031144      MOV    #LLP1,R0     ; DIVISOR = 0
3796 031022 174410      20$:    DIVD  (R0),ACO   ; +N/O = ERROR AND TRAP.
3797 031024 000240              240
3798 031026 170205              STFPS  R5
3799 031030 010537 001176      MOV    R5,$TMP7
3800 031034 170305              STST   R5
3801 031036 010537 001200      MOV    R5,$TMP10
3802 031042 104204              ERROR  204
3803 031044 000447              BR     LLDONE       ; ...AND QUIT.
3804
3805 031046 022626 100200 001172  1$:    CMP    (SP)+,(SP)+ ; TRAP SEEMS OK, FIX STACK.
3806 031050 012737 000004 001174  MOV    #100200,$TMP5 ; EXPECTED FPS...
3807 031056 012737 000004 001174  MOV    #4,$TMP6     ; ...AND FEC.
3808 031064 170205              STFPS  R5
3809 031066 010537 001176      MOV    R5,$TMP7     ; RECEIVED FPS...
3810 031072 170305              STST   R5
3811 031074 010537 001200      MOV    R5,$TMP10    ; ...AND FEC.
3812 031100 004537 004302      JSR    R5,CHECK2
3813 031104 001172 001176      $TMP5,$TMP7
3814 031110 001401              BEQ    3$           ; BR IF BOTH OK.
3815
3816 031112 104203              ERROR  203          ; FPS OR FEC WRONG.
3817 031114 000423              3$:    BR     LLDONE
3818
3819      ; REPORT DIVIDE BY 0 TRAPPED WHEN FID = 1.
3820
3821 031116 170205              LL244: STFPS  R5
3822 031120 010537 001176      MOV    R5,$TMP7
3823 031124 170305              STST   R5
3824 031126 010537 001200      MOV    R5,$TMP10
3825 031132 012637 031142      MOV    (SP)+,1$     ; IN CASE LOOP IS SET.
3826 031136 104205              ERROR  205          ; TRAPPED WHEN IT SHOULDN'T.
3827 031140 000137              JMP    @PC+

```

CJIPAA -- LSI11/23 FPF11 DIAGNOSTIC, PART 1
CJIPAA.P11 28-JAN-81 09:50 T43

MACY11 30G(1063) 28-JAN-81 10:06 PAGE 2-87
DIVD -- FSRC MODE 1, WITH DIVISOR = 0

SEQ 0117

3828 031142 000000 18: 0
3829
3830 031144 000000 000000 000000 LLP1: .WORD 0,0,0,0 : DIVISOR
3831 031154 012345 054321 023456 LLP2: .WORD 12345,54321,23456,76543
3832
3833 031164 LLDONE:
(1) 031164 104412 CLRFP5
(3) 031166 000400 BR TST44

:: CLEAR FP STATUS...
::...AND PROCEED.

3845
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 03117C 000004
3846
3847
3848
3849 031172 004737 031600
3850 031176 000000 0000C0
3851 031202 012345 067012
3852 031206 000000 000000
3853 03:212 000017 000004
3854
3855
3856
3857 031216 004737 031600
3858 031222 065652 125252
3859 031226 065600 000000
3860 031232 040252 125252
3861 031236 000017 000000
3862
3863
3864
3865 031242 004737 031600
3866 031246 076400 000000
3867 031252 076400 000000
3868 031256 040200 000000
3869 031262 000017 000000
3870
3871
3872
3873 031266 004737 031600
3874 031272 056777 177777
3875 031276 054200 000000
3876 031302 042777 177777
3877 031306 000017 000000
3878
3879
3880
3881 031312 004737 031600
3882 031316 012377 177777
3883 031322 012300 000000
3884 031326 040252 125252
3885 031332 000017 000000
3886
3887
3888
3889 031336 004737 031600

```
*****
:TEST 44      DIVF -- FSRC MODE 1
:
:TEST THE DIVF INSTRUCTION WITH A VARIETY OF OPERANDS.
:A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE
:INSTRUCTION, AND CHECK THE RESULTS.
:NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE
:OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE
:OFFENDING INSTRUCTION ITSELF.
:
:*****
TST44: SCOPE
:
:CHECK DIVF WITH (AC 0).
MM1:  JSR      PC,MMSUB
      .WORD    0,0           :AC
      .WORD    12345,67012  :FSRC
      .WORD    0,0           :EXP RESULT.
      .WORD    17,4         :FPS BEFORE AND AFTER.
:
:TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.
MM2:  JSR      PC,MMSUB
      .WORD    65652,125252 :AC
      .WORD    65600,0       :FSRC
      .WORD    40252,125252  :RES
      .WORD    17,0         :FPS BEFORE AND AFTER.
:
:TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.
MM3:  JSR      PC,MMSUB
      .WORD    76400,0       :AC
      .WORD    76400,0       :FSRC
      .WORD    40200,0       :RES
      .WORD    17,0         :FPS'S.
:
:TEST DIVF WITH BOTH OPERANDS POSITIVE.
MM4:  JSR      PC,MMSUB
      .WORD    56777,177777 :AC
      .WORD    54200,0       :FSRC
      .WORD    42777,177777  :RES
      .WORD    17,0         :FPS'S
:
:TEST THE DIVF INSTRUCTION:
MM5:  JSR      PC,MMSUB
      .WORD    12377,177777 :AC
      .WORD    12300,0       :FSRC
      .WORD    40252,125252  :RES
      .WORD    17,0         :FPS'S
:
:TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
MM6:  JSR      PC,MMSUB
```

3890 031342 064600 000001
3891 031346 066600 000000
3892 031352 036200 000001
3893 031356 000017 000000

.WORD 64600,1 ;AC
.WORD 66600,0 ;FSRC
.WORD 36200,1 ;RES
.WORD 17,0 ;FPS'S

TEST DIVF.

3894
3895
3896
3897 031362 004737 031600
3898 031366 034577 177776
3899 031372 023400 000000
3900 031376 051377 177776
3901 031402 000017 000000

MM7: JSR PC,MMSUB
.WORD 34577,177776 ;AC
.WORD 23400,0 ;FSRC
.WORD 51377,177776 ;RES
.WORD 17,0 ;FPS'S

DIVF TEST.

3902
3903
3904
3905 031406 004737 031600
3906 031412 067652 125252
3907 031416 056500 000000
3908 031422 051343 107070
3909 031426 000017 000000

MM8: JSR PC,MMSUB
.WORD 67652,125252 ;AC
.WORD 56500,0 ;FSRC
.WORD 51343,107070 ;RES
.WORD 17,0 ;FPS'S

DIVF WITH AC NEGATIVE, FSRC NEGATIVE.

3910
3911
3912
3913 031432 004737 031600
3914 031436 140400 000000
3915 031442 140500 000000
3916 031446 040052 125253
3917 031452 000017 000000

MM9: JSR PC,MMSUB
.WORD 140400,0 ;AC
.WORD 140500,0 ;FSRC
.WORD 040052,125253 ;RES
.WORD 17,0 ;FPS'S

DIVF WITH AC NEGATIVE AND FSRC POSITIVE.

3918
3919
3920
3921 031456 004737 031600
3922 031462 160077 000000
3923 031466 040277 000000
3924 031472 160000 000000
3925 031476 000017 000010

MM10: JSR PC,MMSUB
.WORD 160077,0 ;AC
.WORD 40277,0 ;FSRC
.WORD 160000,0 ;RES
.WORD 17,10 ;FPS'S.

DIVF WITH AC POSITIVE AND FSRC NEGATIVE.

3926
3927
3928
3929 031502 004737 031600
3930 031506 040400 000000
3931 031512 140500 000000
3932 031516 140052 125253
3933 031522 000017 000010

MM11: JSR PC,MMSUB
.WORD 40400,0 ;AC
.WORD 140500,0 ;FSRC
.WORD 140052,125253 ;RES
.WORD 17,10 ;FPS'S

TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.

3934
3935
3936
3937 031526 004737 031600
3938 031532 060100 000001
3939 031536 040300 000000
3940 031542 060000 000000
3941 031546 000057 000040

MM12: JSR PC,MMSUB
.WORD 60100,1 ;AC
.WORD 40300,0 ;FSRC
.WORD 60000,0 ;RES
.WORD 57,40 ;FPS'S

DIVF WITH POSITIVE OPERANDS AND ROUND MODE.

3942
3943
3944
3945 031552 004737 031600

MM13: JSR PC,MMSUB


```

3946 031556 060100 000001
3947 031562 040300 000000
3948 031566 060000 000001
3949 031572 000017 000000
3950
3951 031576 000471
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968 031600 022637 001164
3969 031604 104411
3970 031606 013701 001164
3971 031612 170011
3972 031614 172427 000000
3973 031620 016100 000014
3974 031624 170100
3975 031626 010100
3976 031630 172410
3977 031632 010037 001172
3978 031636 010100
3979 031640 062700 000004
3980 031644 010037 001174
3981 031650 174410
3982 031652 000240
3983 031654 170204
3984 031656 010437 001166
3985 031662 016137 000016 001170
3986 031670 170011
3987 031672 012700 031752
(1) 031676 174010
3988 031700 010037 001200
3989 031704 010100
3990 031706 062700 000010
3991 031712 010037 001176
3992 031716 010037 031726
3993 031722 004537 004302
3994 031726 000000 031752
3995 031732 001004
3996 031734 023737 001166 001170
3997 031742 001401
3998 031744 104206
3999
4000 031746 000161 000020
  
```

```

.WORD 60100,1 ;AC
.WORD 40300,0 ;FSRC
.WORD 60000,1 ;RES
.WORD 17,0 ;FPS'S

BR MMDONE ; WHEW !!!!!

: SUBROUTINE TO SET-UP, EXECUTE, AND CHECK THE RESULTS OF DIVF.
: CALL:
: JSR PC,@MMMSUB
: .WORD X,X ; AC OPERAND (DIVIDEND).
: .WORD X,X ; FSRC OPERAND (DIVISOR).
: .WORD X,X ; EXPECTED RESULT.
: .WORD X,X ; FPS BEFORE AND EXP FPS AFTER.
: ; RETURN TO CALLER+20

: THE OPERANDS ARE SELECTED USING THE CALLERS PC.
: AFTER EXECUTION, THE QUOTIENT, AND THE RESULTING FPS ARE
: TESTED. IF BOTH OK, RETURN.
: OTHERWISE, REPORT ERROR AND RETURN.

MMMSUB: MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC...
LUPERR ; ; LOOP HERE ON ERROR IF SWR9 1.
MOV $TMP2,R1 ; ...AND USE TO GET ARGS.
SETD
LDD #0,ACO ; CLEAR ACO.
MOV 14(R1),R0
LDFPS R0 ; SET INITIAL FPS (F MODE).
MOV R1,R0
LDF (R0),ACO ; LOAD THE DIVIDEND.
MOV R0,$TMP5
MOV R1,R0
ADD #4,R0 ; POINT TO DIVISOR.
MOV R0,$TMP6
2$: DIVF (R0),ACO ; TEST INSTRUCTION.
NOP
STFPS R4 ; GET THE FPS.
MOV R4,$TMP3
MOV 16(R1),$TMP4 ; EXP FPS.
SETD ; RETURN TO DOUBLE MODE.
MOV #20$,R0
STD ACO,(R0) ; STORE ACO
MOV R0,$TMP10 ; POINTS TO REC'D RESULT.
MOV R1,R0
ADD #10,R0
MOV R0,$TMP7 ; POINTS TO EXP RESULT.
MOV R0,3$
JSR R5,CHECK2 ; RESULT CORRECT ??
3$: 0,20$
BNE 4$ ; BR IF NOT.
CMP $TMP3,$TMP4 ; FPS RIGHT ??
BEQ 5$ ; RETURN IF BOTH OK.
4$: ERROR 206 ; FPS OR RESULT INCORRECT.
5$: JMP 20(R1) ; RETURN
  
```

```
4001  
4002 031752 000000 000000 000000 20$: .WORD 0,0,0,0 ; TMP DATA BUFFER.  
4003  
4004 031762 MMDONE:  
(1) 031762 104412 CLRFP5 ;; CLEAR FP STATUS...  
(3) 031764 000400 BR TST45 ;;...AND PROCEED.
```

4016
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 031766 000004
4017
4018
4019
4020 031770 004737 032542
4021 031774 034277 000000 000000
4022 032004 040277 000000 000000
4023 032014 034200 000000 000000
4024 032024 000217 000200
4025
4026
4027
4028 032030 004737 032542
4029 032034 134277 000000 000000
4030 032044 040277 000000 000000
4031 032054 134200 000000 000000
4032 032064 000217 000210
4033
4034
4035
4036 032070 004737 032542
4037 032074 134200 000000 000000
4038 032104 140300 000000 000000
4039 032114 034200 000000 000000
4040 032124 000257 000240
4041
4042
4043
4044 032130 004737 032542
4045 032134 034300 000000 000000
4046 032144 140300 000000 000000
4047 032154 134200 000000 000000
4048 032164 000217 000210
4049
4050
4051
4052 032170 004737 032542
4053 032174 100400 000000 000000
4054 032204 000500 000000 000000
4055 032214 140052 125252 125252
4056 032224 000257 000250
4057
4058
4059
4060 032230 004737 032542

:TEST 45 DIVD -- FSRC MODE 1

:TEST THE DIVD INSTRUCTION WITH A VARIETY OF OPERANDS.
:A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE
:INSTRUCTION, AND CHECK THE RESULTS.
:NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE
:OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE
:OFFENDING INSTRUCTION ITSELF.

TST45: SCOPE
:DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.
:NN1: JSR PC,NNSUB
:WORD 34277,0,0,0 :AC
:WORD 40277,0,0,0 :FSRC
:WORD 34200,0,0,0 :RES
:WORD 217,200 :FPS BEFORE AND AFTER DIVD.
:DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
:NN2: JSR PC,NNSUB
:WORD 134277,0,0,0 :AC
:WORD 40277,0,0,0 :FSRC
:WORD 134200,0,0,0 :RES
:WORD 217,210 :FPS BEFORE AND AFTER.
:DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
:NN3: JSR PC,NNSUB
:WORD 134300,0,0,1 :AC
:WORD 140300,0,0,0 :FSRC
:WORD 34200,0,0,0 :RES
:WORD 257,240 :FPS BEFORE AND AFTER.
:DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
:NN4: JSR PC,NNSUB
:WORD 34300,0,0,1 :AC
:WORD 140300,0,0,0 :FSRC
:WORD 134200,0,0,1 :RES
:WORD 217,210 :FPS BEFORE AND AFTER.
:DIVD TEST. TRUNCATE.
:NN5: JSR PC,NNSUB
:WORD 100400,0,0,0 :AC
:WORD 500,0,0,0 :FSRC
:WORD 140052,125252,125252,125252 :RES
:WORD 257,250 :FPS'S
:DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
:NN6: JSR PC,NNSUB

```
4061 032234 000400 000000 000000 .WORD 400,0,0,0 ;AC
4062 032244 100500 000000 000000 .WORD 100500,0,0,0 ;FSRC
4063 032254 140052 125252 125252 .WORD 140052,125252,125252,125253 ;RES
4064 032264 000217 000210 .WORD 217,210 ;FPS'S
4065
4066 ;DIVD TEST. ROUND.
4067
4068 032270 004737 032542 NN7: JSR PC,NNSUB
4069 032274 170360 170360 170360 .WORD 170360,170360,170360,170360 ;AC
4070 032304 170360 170360 170360 .WORD 170360,170360,170360,170360 ;FSRC
4071 032314 040200 000000 000000 .WORD 40200,0,0,0 ;RES
4072 032324 000217 000200 .WORD 217,200 ;FPS'S
4073
4074 ; NOW AN OVERFLOW CASE (TWICE).
4075
4076 032330 004737 032532 NN8: JSR PC,NNSUB1
4077 032334 077777 177777 177777 .WORD 077777,-1,-1,-1 ; AC
4078 032344 040000 000000 000000 .WORD 040000,0,0,1 ; FSRC
4079 032354 000177 177777 177777 .WORD 000177,-1,-1,-3 ; RESULT (FIV = 1).
4080 032364 041217 141206 .WORD 41217,141206 ; FPS'S
4081
4082 032370 004737 032532 JSR PC,NNSUB1
4083 032374 077777 177777 177777 .WORD 077777,-1,-1,-1
4084 032404 040000 000000 000000 .WORD 040000,0,0,1
4085 032414 000000 000000 000000 .WORD 0,0,0,0 ; RESULT (FIV = 0).
4086 032424 000217 000206 .WORD 217,206
4087
4088 ; AND FINALLY, AND UNDERFLOW CASE (TWICE).
4089
4090 032430 004737 032532 NN9: JSR PC,NNSUB1
4091 032434 000200 000000 000000 .WORD 000200,0,0,0 ; AC
4092 032444 040200 000000 000000 .WORD 040200,0,0,1 ; FSRC
4093 032454 000177 177777 177777 .WORD 000177,-1,-1,-2 ; RESULT (FIU = 1).
4094 032464 042217 142204 .WORD 42217,142204 ; FPS'S.
4095
4096 032470 004737 032532 JSR PC,NNSUB1
4097 032474 000200 000000 000000 .WORD 000200,0,0,0
4098 032504 040200 000000 000000 .WORD 040200,0,0,1
4099 032514 000000 000000 000000 .WORD 0,0,0,0 ; RESULT (FIU = 0).
4100 032524 000217 000204 .WORD 217,204
4101
4102 032530 000474 BR MNDONE ; FINISHED.
4103
4104 ; SUBROUTINE TO SET-UP, EXECUTE, AND CHECK THE RESULTS OF DIVD.
4105
4106 ; CALL:
4107 JSR PC,NNSUB ; OR NNSUB1
4108 .WORD X,X,X,X ; AC OPERAND (DIVIDEND).
4109 .WORD X,X,X,X ; FSRC OPERAND (DIVISOR).
4110 .WORD X,X,X,X ; EXPECTED RESULT.
4111 .WORD X,X ; FPS BEFORE AND EXP FPS AFTER.
4112 ; RETURN TO CALLER+34
4113
4114 ; THE OPERANDS ARE SELECTED USING THE CALLERS PC.
4115 ; AFTER EXECUTION, THE QUOTIENT, AND THE RESULTING FPS ARE
4116 ; TESTED. IF BOTH OK, RETURN.
```

```
4117 ; OTHERWISE, REPORT ERROR AND RETURN.  
4118 ;  
4119 032532 112737 000040 055740 NNSUB1: MOVB #40,EM207X ; ENABLE...  
4120 032540 000402 SKP2  
4121 032542 105037 055740 NNSUB: CLRB EM207X ;...OR DISABLE MESSAGE EXTENSION.  
4122 032546 012637 001164 MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC...  
4123 032552 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
4124 032554 013701 001164 MOV $TMP2,R1 ;...AND USE TO GET ARGS.  
4125 032560 016100 000030 MOV 30(R1),R0  
4126 032564 170100 LDFPS R0 ; SET INITIAL FPS.  
4127 032566 010100 MOV R1,R0  
4128 032570 172410 LDD (R0),ACO ; LOAD THE DIVIDEND.  
4129 032572 010037 001172 MOV R0,$TMP5  
4130 032576 010100 MOV R1,R0  
4131 032600 062700 000010 ADD #10,R0 ;POINT TO DIVISOR.  
4132 032604 010037 001174 MOV R0,$TMP6  
4133 032610 174410 2$: DIVD (R0),ACO ;TEST INSTRUCTION.  
4134 032612 000240 NOP  
4135 032614 170204 STFPS R4 ;GET THE FPS.  
4136 032616 010437 001166 MOV R4,$TMP3  
4137 032622 016137 000032 001170 MOV 32(R1),$TMP4 ; EXP FPS.  
4138 032630 170011 SETD ; INSURE DOUBLE STILL SET.  
4139 032632 012700 032712 MOV #20$,R0  
(1) 032636 174010 STD ACO,(R0) ;: STORE ACO  
4140 032640 010037 001200 MOV R0,$TMP10 ; POINTS TO REC'D RESULT.  
4141 032644 010100 MOV R1,R0  
4142 032646 062700 000020 ADD #20,R0  
4143 032652 010037 001176 MOV R0,$TMP7 ; POINTS TO EXP RESULT.  
4144 032656 010037 032666 MOV R0,3$  
4145 032662 004537 004312 JSR R5,CHECK4 ; RESULT CORRECT ??  
4146 032666 000000 032712 3$: O,20$  
4147 032672 001004 BNE 4$ ; BR IF NOT.  
4148 032674 023737 001166 001170 CMP $TMP3,$TMP4 ; FPS RIGHT ??  
4149 032702 001401 BEQ 5$ ; RETURN IF BOTH OK.  
4150 032704 104207 4$: ERROR 207 ; FPS OR RESULT INCORRECT.  
4151  
4152 032706 000161 000034 5$: JMP 34(R1) ; RETURN  
4153  
4154 032712 000000 000000 000000 20$: .WORD 0,0,0,0 ; TMP DATA BUFFER.  
4155  
4156 032722 NNDONE:  
(1) 032722 114412 CLRFPS ;: CLEAR FP STATUS...  
(3) 032724 000400 BR TST46 ;:...AND PROCEED.
```

4168
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 032726 000004
4169
4170
4171
4172 032730 004737 033406
4173 032734 000000 000000
4174 032740 000000 000000
4175 032744 000000 000000
4176 032750 000017 000004
4177
4178
4179
4180 032754 004737 033406
4181 032760 071625 034435
4182 032764 000000 000000
4183 032770 000000 000000
4184 032774 000017 000004
4185
4186
4187
4188 033000 004737 033406
4189 033004 000000 000000
4190 033010 071625 153443
4191 033014 000000 000000
4192 033020 000017 000004
4193
4194
4195
4196 033024 004737 033406
4197 033030 040200 000000
4198 033034 040177 177777
4199 033040 040177 177777
4200 033044 000017 000000
4201
4202
4203
4204 033050 004737 033406
4205 033054 040177 177777
4206 033060 040200 000000
4207 033064 040177 177777
4208 033070 000057 000040
4209
4210
4211
4212 033074 004737 033406

```
.....  
:TEST 46      MULF -- FSRC MODE 1  
:  
:TEST THE MULF INSTRUCTION WITH A VARIETY OF OPERANDS.  
:A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE  
:INSTRUCTION, AND CHECK THE RESULTS.  
:NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE  
:OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE  
:OFFENDING INSTRUCTION ITSELF.  
:.....  
TST46: SCOPE  
:  
:MULF WITH (FSRC=AC=0)  
:  
PP1:   JSR      PC,PPSUB  
       .WORD    0,0           ;AC  
       .WORD    0,0           ;FSRC  
       .WORD    0,0           ;RES  
       .WORD    17,4        ; FPS BEFORE AND AFTER MULF.  
:  
:MULF WITH (FSRC=0).  
:  
PP2:   JSR      PC,PPSUB  
       .WORD    71625,34435  ;AC  
       .WORD    0,0           ;FSRC  
       .WORD    0,0           ;RES  
       .WORD    17,4        ; FPS BEFORE AND AFTER.  
:  
:MULF WITH (AC=0)  
:  
PP3:   JSR      PC,PPSUB  
       .WORD    0,0           ;AC  
       .WORD    071625,153443 ;FSRC  
       .WORD    0,0           ;RES  
       .WORD    17,4        ; FPS'S  
:  
:MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.  
:  
PP4:   JSR      PC,PPSUB  
       .WORD    40200,0       ;AC  
       .WORD    40177,-1     ;FSRC  
       .WORD    40177,-1     ;RES  
       .WORD    17,0        ; FPS'S.  
:  
:MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.  
:  
PP5:   JSR      PC,PPSUB  
       .WORD    40177,-1     ;AC  
       .WORD    40200,0       ;FSRC  
       .WORD    40177,-1     ;RES  
       .WORD    57,40       ; FPS'S  
:  
:MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.  
:  
PP6:   JSR      PC,PPSUB
```

4213 033100 040100 000000
4214 033104 040100 000000
4215 033110 040020 000000
4216 033114 000017 000000

.WORD 40100,0 ;AC
.WORD 40100,0 ;FSRC
.WORD 40020,0 ;RES
.WORD 17,0 ;FPS'S

PP7: ;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.

4220 033120 004737 033406
4221 033124 017500 000000
4222 033130 023652 125252
4223 033134 003177 177777
4224 033140 000017 000000

PP7: JSR PC,PPSUB
.WORD 17500,0 ;AC
.WORD 23652,125252 ;FSRC
.WORD 3177,-1 ;RES
.WORD 17,0 ;FPS'S

PP8: ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.

4228 033144 004737 033406
4229 033150 040342 000000
4230 033154 176542 000000
4231 033160 176707 102000
4232 033164 000017 000010

PP8: JSR PC,PPSUB
.WORD 40342,0 ;AC
.WORD 176542,0 ;FSRC
.WORD 176707,102000 ;RES
.WORD 17,10 ;FPS'S

PP9: ;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.

4236 033170 004737 033406
4237 033174 140200 000000
4238 033200 007417 007417
4239 033204 107417 007417
4240 033210 000017 000010

PP9: JSR PC,PPSUB
.WORD 140200,0 ;AC
.WORD 7417,7417 ;FSRC
.WORD 107417,7417 ;RES
.WORD 17,10 ;FPS'S

PP10: ;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.

4244 033214 004737 033406
4245 033220 144600 000000
4246 033224 154000 000000
4247 033230 060400 000000
4248 033234 000017 000000

PP10: JSR PC,PPSUB
.WORD 144600,0 ;AC
.WORD 154000,0 ;FSRC
.WORD 60400,0 ;RES
.WORD 17,0 ;FPS'S

PP11: ;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.

4252 033240 004737 033406
4253 033244 140300 000000
4254 033250 160000 000001
4255 033254 060100 000002
4256 033260 000017 000000

PP11: JSR PC,PPSUB
.WORD 140300,0 ;AC
.WORD 160000,1 ;FSRC
.WORD 60100,2 ;RES
.WORD 17,0 ;FPS'S

PP12: ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.

4260 033264 004737 033406
4261 033270 060000 000001
4262 033274 140300 000000
4263 033300 160100 000001
4264 033304 000057 000050

PP12: JSR PC,PPSUB
.WORD 60000,1 ;AC
.WORD 140300,0 ;FSRC
.WORD 160100,1 ;RES
.WORD 57,50 ;FPS'S

PP13: ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.

4268 033310 004737 033406

PP13: JSR PC,PPSUB

4269 033314 040277 000000
 4270 033320 060000 000001
 4271 033324 060077 000001
 4272 033330 000017 000000
 4273
 4274
 4275
 4276 033334 004737 033406
 4277 033340 060252 125252
 4278 033344 060100 000000
 4279 033350 000177 177777
 4280 033354 041017 141006
 4281
 4282 033360 004737 033406
 4283 033364 020052 125252
 4284 033370 020300 000000
 4285 033374 000177 177777
 4286 033400 042017 142004
 4287
 4288 033404 000471
 4289
 4290
 4291
 4292
 4293
 4294
 4295
 4296
 4297
 4298
 4299
 4300
 4301
 4302
 4303
 4304
 4305 033406 012637 001164
 4306 033412 104411
 4307 033414 013701 001164
 4308 033420 170011
 4309 033422 172427 000000
 4310 033426 016100 000014
 4311 033432 170100
 4312 033434 010100
 4313 033436 172410
 4314 033440 010037 001172
 4315 033444 010100
 4316 033446 062700 000004
 4317 033452 010037 001174
 4318 033456 171010
 4319 033460 000240
 4320 033462 170204
 4321 033464 010437 001166
 4322 033470 016137 000016 001170
 4323 033476 170011
 4324 033500 012700 033560

```

        .WORD 40277,0      ;AC
        .WORD 60000,1     ;FSRC
        .WORD 60077,1     ;RES
        .WORD 17,0       ;FPS'S
    :
    : OVER AND UNDERFLOW (1 EACH).
PP14: JSR PC,PPSUB
        .WORD 060252,125252 ; AC
        .WORD 060100,0     ; FSRC
        .WORD 000177,-1   ; FPS'S (FIV = 1).
        .WORD 41017,141006
    JSR PC,PPSUB
        .WORD 020052,125252 ; AC
        .WORD 020300,0     ; FSRC
        .WORD 000177,-1   ; FPS'S (FIU = 1).
        .WORD 42017,142004
    BR PPDONE ; FINISHED.
    :
    : SUBROUTINE TO SET-UP, EXECUTE, AND CHECK THE RESULTS OF MULF.
    :
    : CALL:
    : JSR PC,PPSUB
    : .WORD X,X ; AC OPERAND (MULTIPLICAND).
    : .WORD X,X ; FSRC OPERAND (MULTIPLIER).
    : .WORD X,X ; EXPECTED PRODUCT.
    : .WORD X,X ; FPS BEFORE AND EXP FPS AFTER.
    : ; RETURN TO CALLER+20
    :
    : THE OPERANDS ARE SELECTED USING THE CALLERS PC.
    : AFTER EXECUTION, THE PRODUCT, AND THE RESULTING FPS ARE
    : TESTED. IF BOTH OK, RETURN.
    : OTHERWISE, REPORT ERROR AND RETURN.
PPSUB: MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC...
        LUPERR ; ; LOOP HERE ON ERROR IF SWR9 - 1.
        MOV $TMP2,R1 ; ...AND USE AS ARG POINTER.
        SETD
        LDD #0,ACO ; CLEAR ACO.
        MOV 14(R1),R0
        LDFPS R0 ; SET INITIAL FPS (F MODE).
        MOV R1,R0
        LDF (R0),ACO ; LOAD THE MULTIPLICAND.
        MOV R0,$TMP5
        MOV R1,R0
        ADD #4,R0 ; POINT TO MULTIPLIER.
        MOV R0,$TMP6
    2$: MULF (R0),ACO ; TEST INSTRUCTION.
        NOP
        STFPS R4 ; GET THE FPS.
        MOV R4,$TMP3
        MOV 16(R1),$TMP4 ; EXP FPS.
        SETD ; RETURN TO DOUBLE MODE.
        MOV #20$,R0
    
```


(1)	033504	174010				STD	AC0,(R0)	:: STORE AC0
4325	033506	010037	001200			MOV	R0,\$TMP10	; POINTS TO REC'D RESULT.
4326	033512	010100				MOV	R1,R0	
4327	033514	062700	000010			ADD	#10,R0	
4328	033520	010037	001176			MOV	R0,\$TMP7	; POINTS TO EXP RESULT.
4329	033524	010037	033534			MOV	R0,3\$	
4330	033530	004537	004302			JSR	R5,CHECK2	; RESULT CORRECT ??
4331	033534	000000	033560	3\$:		0,20\$		
4332	033540	001004				BNE	4\$; BR IF NOT.
4333	033542	023737	001166	001170		CMP	\$TMP3,\$TMP4	; FPS RIGHT ??
4334	033550	001401				BEQ	5\$; RETURN IF BOTH OK.
4335	033552	104210			4\$:	ERROR	210	; FPS OR RESULT INCORRECT.
4336								
4337	033554	000161	000020		5\$:	JMP	20(R1)	; RETURN
4338								
4339	033560	000000	000000	000000	20\$:	.WORD	0,0,0,0	; TMP DATA BUFFER.
4340								
4341	033570				PPDONE:			
(1)	033570	104412				CLRFPS		:: CLEAR FP STATUS...
3)	033572	000400				BR	1ST47	::...AND PROCEED.

4353
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 033574 000004
4354
4355
4356
4357 033576 004737 034250
4358 033602 040200 000000 000000
4359 033612 023777 177777 177777
4360 033622 023777 177777 177777
4361 033632 000217 000200
4362
4363
4364
4365 033636 004737 034250
4366 033642 065400 000000 000000
4367 033652 037577 177777 177777
4368 033662 064777 177777 177777
4369 033672 000257 000240
4370
4371
4372
4373 033676 004737 034250
4374 033702 137577 177777 177777
4375 033712 165400 000000 000000
4376 033722 065000 000000 000000
4377 033732 000217 000200
4378
4379
4380
4381 033736 004737 034250
4382 033742 017500 000000 000000
4383 033752 123652 125252 125252
4384 033762 103177 177777 177777
4385 033772 000217 000210
4386
4387
4388
4389 033776 004737 034240
4390 034002 060252 125252 125252
4391 034012 160300 000000 000000
4392 034022 100377 177777 177777
4393 034032 041217 141212
4394
4395
4396
4397 034036 004737 034240

:TEST 47 MULD -- FSRC MODE 1
:
:TEST THE MULD INSTRUCTION WITH A VARIETY OF OPERANDS.
:A COMMON SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE
:INSTRUCTION, AND CHECK THE RESULTS.
:NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THE PC OF THE
:OPERAND SET WHICH RESULTED IN THE ERROR AND NOT THAT OF THE
:OFFENDING INSTRUCTION ITSELF.
:
:*****
TST47: SCOPE
:
:MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.
:Q01: JSR PC,QQSUB
:WORD 40200,0,0,0 :AC
:WORD 23777,-1,-1,-1 :FSRC
:WORD 23777,-1,-1,-1 :RES
:WORD 217,200 :FPS BEFORE AND AFTER MULD.
:
:MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
:Q02 JSR PC,QQSUB
:WORD 65400,0,0,1 :AC
:WORD 37577,-1,-1,-2 :FSRC
:WORD 64777,-1,-1,-1 :RES
:WORD 257,240 :FPS'S.
:
:MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
:Q03: JSR PC,QQSUB
:WORD 137577,-1,-1,-2 :AC
:WORD 165400,0,0,1 :FSRC
:WORD 65000,0,0,0 :RES
:WORD 217,200 :FPS'S.
:
:MULD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
:Q04: JSR PC,QQSUB
:WORD 17500,0,0,0 :AC
:WORD 123652,125252,125252,125252 :FSRC
:WORD 103177,-1,-1,-1 :RES
:WORD 217,210 :FPS'S.
:
: OVERFLOW -- FINAL EXP > 128.
:Q05: JSR PC,QQSUB1
:WORD 060252,125252,125252,125252 :AC
:WORD 160300,0,0,0 :FSRC
:WORD 100377,-1,-1,-1 :RESULT (FIV - 1).
:WORD 41217,141212 :FPS'S.
:
: OVERFLOW -- FINAL EXP = 128.
:Q06: JSR PC,QQSUB1

4485
 (3)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2) 034434 000004
 4486
 4487
 4488
 4489 034436 004737 035160
 4490 034442 000000 000000
 4491 034446 000000 000000
 4492 034452 000000 000000
 4493 034456 000000 000000
 4494 034462 000017 000004
 4495
 4496
 4497
 4498 034466 004737 035160
 4499 034472 123456 076543
 4500 034476 000000 000000
 4501 034502 000000 000000
 4502 034506 000000 000000
 4503 034512 000017 000004
 4504
 4505
 4506
 4507 034516 004737 035160
 4508 034522 000000 000000
 4509 034526 076543 021234
 4510 034532 000000 000000
 4511 034536 000000 000000
 4512 034542 000017 000004
 4513
 4514
 4515
 4516 034546 004737 035160
 4517 034552 046252 125252
 4518 034556 040300 000000
 4519 034562 000000 000000
 4520 034566 046377 177777
 4521 034572 000017 000004
 4522
 4523
 4524
 4525 034576 004737 035160
 4526 034602 077652 125252
 4527 034606 040300 000000
 4528 034612 000000 000000
 4529 034616 077777 177777
 4530 034622 000017 000004
 4531
 4532
 4533

```

:*****
:TEST 50      MODF -- FSRC MODE 1
:
:THIS IS A TEST OF THE MODF INSTRUCTION.
:A COMMON SUBROUTINE IS USED TO EXECUTE THE TEST AS BEFORE.
:
:*****
TST50: SCOPE
:
:MODF WITH (FSRC=AC=0)
:
TT1:  JSR      PC,TTSUB
      .WORD    0,0          :AC
      .WORD    0,0          :FSRC
      .WORD    0,0          :FRACTIONAL RES.
      .WORD    0,0          :INTEGER RES.
      .WORD    17,4         :FPS BEFORE AND AFTER MODF.
:
:MODF TEST, WITH (FSRC=0)
:
TT2:  JSR      PC,TTSUB
      .WORD    123456,76543 :AC
      .WORD    0,0          :FSRC
      .WORD    0,0          :FRACTIONAL RES.
      .WORD    0,0          :INTEGER RESULT.
      .WORD    17,4         :FPS'S
:
:MODF TEST WITH (AC=0)
:
TT3:  JSR      PC,TTSUB
      .WORD    0,0          :AC
      .WORD    76543,21234  :FSRC
      .WORD    0,0          :FRACTIONAL RES.
      .WORD    0,0          :INTEGER RES.
      .WORD    17,4         :FPS'S
:
:MODF TEST WITH EXPONENT OF THE RESULT = 25
:
TT4:  JSR      PC,TTSUB
      .WORD    46252,125252 :AC
      .WORD    40300,0       :FSRC
      .WORD    0,0          :FRACTIONAL RES.
      .WORD    46377,-1     :INTEGER RES.
      .WORD    17,4         :FPS'S.
:
:MODF TEST WITH EXPONENT OF THE RESULT = 127
:
TT5:  JSR      PC,TTSUB
      .WORD    77652,125252 :AC
      .WORD    40300,0       :FSRC
      .WORD    0,0          :FRACTIONAL RES.
      .WORD    77777,-1     :INTEGER RES.
      .WORD    17,4         :FPS'S
:
:MODF TEST WITH EXPONENT OF RESULT = 25
:
:*** THIS ONE FAILS UNDER MIMIC ???? ***
:

```

4534
4535 034626 004737 035160
4536 034632 046200 000001
4537 034636 040340 000000
4538 034642 000000 000000
4539 034646 046340 000001
4540 034652 000017 000004
4541
4542
4543
4544
4545 034656 004737 035160
4546 034662 046000 000001
4547 034666 040340 000000
4548 034672 040100 000000
4549 034676 046140 000001
4550 034702 000017 000000
4551
4552
4553
4554 034706 004737 035160
4555 034712 042577 177777
4556 034716 040200 000000
4557 034722 040177 176000
4558 034726 042577 140000
4559 034732 000017 000000
4560
4561
4562
4563 034736 004737 035160
4564 034742 042577 140001
4565 034746 040200 000000
4566 034752 034600 000000
4567 034756 042577 140000
4568 034762 000017 000000
4569
4570
4571
4572 034766 004737 035160
4573 034772 042377 100000
4574 034776 040200 000000
4575 035002 000000 000000
4576 035006 042377 100000
4577 035012 000017 000004
4578
4579
4580
4581 035016 004737 035160
4582 035022 040177 177777
4583 035026 040200 000000
4584 035032 040177 177777
4585 035036 000000 000000
4586 035042 000017 000000
4587
4588
4589

TT6: JSR PC,TTSUB
.WORD 46200,1 ;AC
.WORD 40340,0 ;FSRC
.WORD 0,0 ;FRACTIONAL RES.
.WORD 46340,1 ;INTEGER RES.
.WORD 17,4
:MODF TEST WITH EXPONENT OF THE RESULT = 24
*** THIS ONE TOO, SOMFTIMES ??? ***
TT7: JSR PC,TTSUB
.WORD 46000,1 ;AC
.WORD 40340,0 ;FSRC
.WORD 40100,0 ;FRACTIONAL RES.
.WORD 46140,1 ;INTEGER RESULT.
.WORD 17,0 ;FPS'S
:MODF TEST WITH EXPONENT OF THE RESULT = 10
TT8: JSR PC,TTSUB
.WORD 42577,-1 ;AC
.WORD 40200,0 ;FSRC
.WORD 40177,176000 ;FRACTIONAL RES.
.WORD 42577,140000 ;INTEGER RES.
.WORD 17,0 ;FPS'S
:MODF TEST WITH THE EXPONENT OF THE RESULT = 10
TT9: JSR PC,TTSUB
.WORD 42577,140001 ;AC
.WORD 40200,0 ;FSRC
.WORD 34600,0 ;FRACTIONAL RES.
.WORD 42577,140000 ;INTEGER RES.
.WORD 17,0 ;FPS'S
:MODF TEST WITH EXPONENT OF THE RESULT = 9
TT10: JSR PC,TTSUB
.WORD 42377,100000 ;AC
.WORD 40200,0 ;FSRC
.WORD 0,0 ;FRACTIONAL RES.
.WORD 42377,100000 ;INTEGER RES.
.WORD 17,4 ;FPS'S
:MODF TEST WITH EXPONENT OF THE RESULT = 0
TT11: JSR PC,TTSUB
.WORD 40177,-1 ;AC
.WORD 40200,0 ;FSRC
.WORD 40177,-1 ;FRACTIONAL RES.
.WORD 0,0 ;INTEGER RES.
.WORD 17,0 ;FPS'S
:MODF TEST WITH EXPONENT OF THE RESULT = -15

4590 035046 004737 035160
 4591 035052 034377 177777
 4592 035056 040200 000000
 4593 035062 034377 177777
 4594 035066 000000 000000
 4595 035072 000017 000000
 4596
 4597
 4598
 4599 035076 004737 035160
 4600 035102 020000 000001
 4601 035106 040300 000000
 4602 035112 020100 000002
 4603 035116 000000 000000
 4604 035122 000017 000000
 4605
 4606
 4607
 4608 035126 004737 035160
 4609 035132 142777 170000
 4610 035136 040200 000000
 4611 035142 140000 000000
 4612 035146 142777 160000
 4613 035152 000017 000010
 4614
 4615 035156 000517
 4616
 4617
 4618
 4619
 4620
 4621
 4622
 4623
 4624
 4625
 4626
 4627
 4628
 4629
 4630
 4631
 4632
 4633 035160 012637 001164
 4634 035164 104411
 4635 035166 013701 001164
 4636 035172 170011
 4637 035174 172427 000000
 4638 035200 172527 000000
 4639 035204 016100 000020
 4640 035210 170100
 4641 035212 010100
 4642 035214 172410
 4643 035216 010037 001172
 4644 035222 010100
 4645 035224 062700 000004

```

TT12: JSR PC,TTSUB
      .WORD 34377,-1      ;AC
      .WORD 40200,0      ;FSRC
      .WORD 34377,-1      ;FRACTIONAL RES.
      .WORD 0,0          ;INTEGER RES.
      .WORD 17,0         ;FPS'S
      ;
      ;MODF TEST WITH EXPONENT OF RESULT = -64, IN ROUND MODE
TT13: JSR PC,TTSUB
      .WORD 20000,1      ;AC
      .WORD 40300,0      ;FSRC
      .WORD 20100,2      ;FRACTIONAL RES.
      .WORD 0,0          ;INTEGER RES.
      .WORD 17,0         ;FPS'S
      ;
      ;MODF TEST WITH EXPONENT OF RESULT = 11
TT14: JSR PC,TTSUB
      .WORD 142777,170000 ;AC
      .WORD 40200,0      ;FSRC
      .WORD 140000,0     ;FRACTIONAL RES.
      .WORD 142777,160000 ;INTEGER RES.
      .WORD 17,10       ;FPS'S
      ;
      BR TTDONE          ; FINISHED.
      ;
      ; SUBROUTINE TO SET-UP, EXECUTE, AND CHECK THE RESULTS OF MODF.
      ;
      ; CALL:
      ;
      ; JSR PC,TTSUB
      ; .WORD X,X      ; AC OPERAND (MULTIPLICAND).
      ; .WORD X,X      ; FSRC OPERAND (MULTIPLIER).
      ; .WORD X,X      ; RESULTING FRACTION (AC0).
      ; .WORD X,X      ; RESULTING INTEGER (AC1).
      ; .WORD X,X      ; FPS BEFORE AND EXP FPS AFTER.
      ; .WORD X,X      ; RETURN TO CALLER+24
      ;
      ; THE OPERANDS ARE SELECTED USING THE CALLERS PC.
      ; AFTER EXECUTION THE RESULTING INTEGER, FRACTION, AND FPS
      ; ARE TESTED. IF ALL OK, RETURN.
      ; OTHERWISE, REPORT ERROR AND RETURN.
      ;
      ; TSUB: MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC...
      ; LUPERR ; ; LOOP HERE ON ERROR IF SWR9 - 1.
      ; MOV $TMP2,R1 ; ...AND USE AS ARG POINTER.
      ; SETD
      ; LDD #0,AC0 ; CLEAR AC0.
      ; LDD #0,AC1 ; AND AC1.
      ; MOV 20(R1),R0
      ; LDFPS R0 ; SET INITIAL FPS (F MODE).
      ; MOV R1,R0
      ; LDF (R0),AC0 ; LOAD THE MULTIPLICAND.
      ; MOV R0,$TMP5
      ; MOV R1,R0
      ; ADD #4,R0 ; POINT TO MULTIPLIER.
  
```

4646	035230	010037	001174			MOV	R0,\$TMP6		
4647	035234	171410		2\$:		MODF	(R0),AC0	:	TEST INSTRUCTION.
4648	035236	000240				NOP			
4649	035240	170204				STFPS	R4	:	GET THE FPS.
4650	035242	010437	001166			MOV	R4,\$TMP3		
4651	035246	016137	000022	001170		MOV	22(R1),\$TMP4	:	EXP FPS.
4652	035254	170011				SETD			
4653	035256	012700	035376			MOV	#20\$,R0		
(1)	035262	174010				STD	AC0,(R0)	::	STORE AC0
4654	035264	010037	001200			MOV	R0,\$TMP10	::	...AND SET POINTER.
4655	035270	012700	035406			MOV	#21\$,R0		
(1)	035274	174110				STD	AC1,(R0)	::	STORE AC1
4656	035276	010037	001204			MOV	R0,\$TMP12	::	...AND SET POINTER.
4657	035302	010100				MOV	R1,R0		
4658	035304	062700	000010			ADD	#10,R0		
4659	035310	010037	001176			MOV	R0,\$TMP7	:	POINTS TO EXP FRACTION.
4660	035314	010037	035340			MOV	R0,3\$:	IN THE CHECKER TOO.
4661	035320	062700	000004			ADD	#4,R0		
4662	035324	010037	001202			MOV	R0,\$TMP11	:	...AND EXP INTEGER.
4663	035330	010037	035352			MOV	R0,4\$		
4664	035334	004537	004302			JSR	R5,CHECK2	:	FRACTION CORRECT ??
4665	035340	000000	035376		3\$:	0,20\$			
4666	035344	001011				BNE	5\$:	BR IF NOT.
4667	035346	004537	004302			JSR	R5,CHECK2	:	YES, INTEGER CORRECT ??
4668	035352	000000	035406		4\$:	0,21\$			
4669	035356	001004				BNE	5\$:	BR IF NOT.
4670	035360	023737	001166	001170		CMP	\$TMP3,\$TMP4	:	FPS RIGHT ??
4671	035366	001401				BEQ	6\$:	RETURN IF ALL OK.
4672									
4673	035370	104220			5\$:	ERROR	220	:	FPS WRONG.
4674									
4675	035372	000161	000024		6\$:	JMP	24(R1)	:	RETURN
4676									
4677	035376	000000	000000	000000	20\$:	.WORD	0,0,0,0	:	FRACTION BUFFER.
4678	035406	000000	000000	000000	21\$:	.WORD	0,0,0,0	:	INTEGER BUFFER.
4679									
4680	035416					TTDONE:			
(1)	035416	104412				CLRFP		::	CLEAR FP STATUS...
(3)	035420	000400				BR	TST51	::	...AND PROCEED.

4688
(3)
(4)
(4)
(4)
(4)
(3)
(2) 035422 000004
4689
4690
4691
4692 035424 004737 037026
4693 035430 000000 000000 000000
4694 035440 000000 000000 000000
4695 035450 000000 000000 000000
4696 035460 000000 000000 000000
4697 035470 000217 000204
4698
4699
4700
4701 035474 004737 037026
4702 035500 012345 067012 034567
4703 035510 000000 000000 000000
4704 035520 000000 000000 000000
4705 035530 000000 000000 000000
4706 035540 000217 000204
4707
4708
4709
4710 035544 004737 037026
4711 035550 000000 000000 000000
4712 035560 072727 127272 072727
4713 035570 000000 000000 000000
4714 035600 000000 000000 000000
4715 035610 000217 000204
4716
4717
4718
4719 035614 004737 037026
4720 035620 056252 125252 125252
4721 035630 040300 000000 000000
4722 035640 000000 000000 000000
4723 035650 056377 177777 177777
4724 035660 000217 000204
4725
4726
4727
4728 035664 004737 037026
4729 035670 140240 000000 000000
4730 035700 063714 146314 133572
4731 035710 000000 000000 000000
4732 035720 163777 177777 162531
4733 035730 000217 000204
4734
4735
4736

```
*****
:TEST 51      MODD -- FSRC MODE 1
:
:THIS IS A TEST OF THE MODD INSTRUCTION.
:*A COMMON SUBROUTINE IS USED TO EXECUTE THE TEST AS BEFORE.
:
:*****
TST51: SCOPE
:
:MODD WITH (FSRC=AC=0)
:
UU1:  JSR      PC,UUSUB
      .WORD    0,0,0,0          :AC
      .WORD    0,0,0,0          :FSRC
      .WORD    0,0,0,0          :FRACTIONAL RES.
      .WORD    0,0,0,0          :INTEGER RES.
      .WORD    217,204         :FPS BEFORE AND AFTER MODD.
:
:MODD TEST WITH FSRC=0
:
UU2:  JSR      PC,UUSUB
      .WORD    012345,67012,34567,012345 :AC
      .WORD    0,0,0,0          :FSRC
      .WORD    0,0,0,0          :FRACTIONAL RES.
      .WORD    0,0,0,0          :INTEGER RES.
      .WORD    217,204         :FPS'S
:
:MODD TEST WITH (AC=0)
:
UU3:  JSR      PC,UUSUB
      .WORD    0,0,0,0          :AC
      .WORD    72727,127272,72727,127272 :FSRC
      .WORD    0,0,0,0          :FRACTIONAL RES.
      .WORD    0,0,0,0          :INTEGER RES.
      .WORD    217,204         :FPS'S
:
:MODD TEST WITH EXPONENT OF THE RESULT = 57
:
UU4:  JSR      PC,UUSUB
      .WORD    56252,125252,125252,125250 :AC
      .WORD    40300,0,0,0          :FSRC
      .WORD    0,0,0,0          :FRACTIONAL RES.
      .WORD    56377,-1,-1,-4      :INTEGER RES.
      .WORD    217,204         :FPS'S
:
:MODD TEST WITH EXPONENT OF THE RESULT = 79
:
UU5:  JSR      PC,UUSUB
      .WORD    140240,0,0,0          :AC
      .WORD    63714,146314,133572,167737 :FSRC
      .WORD    0,0,0,0          :FRACTIONAL RES.
      .WORD    163777,-1,162531,125726 :INTEGER RES.
      .WORD    217,204         :FPS'S
:
:MODD TEST WITH EXPONENT OF THE RESULT = 57
:
```

4737	035734	004737	037026		UU6:	JSR	PC,UUSUB		
4738	035740	056200	000000	000000		.WORD	56200,0,0,1		:AC
4739	035750	040340	000000	000000		.WORD	40340,0,0,0		:FSRC
4740	035760	000000	000000	000000		.WORD	0,0,0,0		:FRACTIONAL RES.
4741	035770	056340	000000	000000		.WORD	56340,0,0,1		:INTEGER RES.
4742	036000	000217	000204			.WORD	217,204		:FPS'S
4743									
4744									
4745									
4746	036004	004737	037026		UU7:	JSR	PC,UUSUB		
4747	036010	056000	000000	000000		.WORD	56000,0,0,1		:AC
4748	036020	040340	000000	000000		.WORD	40340,0,0,0		:FSRC
4749	036030	040100	000000	000000		.WORD	40100,0,0,0		:FRACTIONAL RES.
4750	036040	056140	000000	000000		.WORD	56140,0,0,1		:INTEGER RES.
4751	036050	000217	000200			.WORD	217,200		:FPS'S
4752									
4753									
4754									
4755	036054	004737	037026		UU8:	JSR	PC,UUSUB		
4756	036060	051177	177777	177777		.WORD	51177,-1,-1,-1		:AC
4757	036070	040200	000000	000000		.WORD	40200,0,0,0		:FSRC
4758	036100	040177	177760	000000		.WORD	40177,-20,0,0		:FRACTIONAL RES.
4759	036110	051177	177777	177760		.WORD	51177,-1,-20,0		:INTEGER RES.
4760	036120	000217	000200			.WORD	217,200		:FPS'S
4761									
4762									
4763									
4764	036124	004737	037026		UU9:	JSR	PC,UUSUB		
4765	036130	040200	000000	000000		.WORD	40200,0,0,0		:AC
4766	036140	047577	177777	176000		.WORD	47577,-1,176000,1		:FSRC
4767	036150	031600	000000	000000		.WORD	31600,0,0,0		:FRACTIONAL RES.
4768	036160	047577	177777	176000		.WORD	47577,-1,176000,0		:INTEGER RES.
4769	036170	000217	000200			.WORD	217,200		:FPS'S
4770									
4771									
4772									
4773	036174	004737	037026		UU10:	JSR	PC,UUSUB		
4774	036200	047777	177777	177000		.WORD	47777,-1,177000,0		:AC
4775	036210	040200	000000	000000		.WORD	40200,0,0,0		:FSRC
4776	036220	000000	000000	000000		.WORD	0,0,0,0		:FRACTIONAL RES.
4777	036230	047777	177777	177000		.WORD	47777,-1,177000,0		:INTEGER RES.
4778	036240	000217	000204			.WORD	217,204		:FPS'S
4779									
4780									
4781									
4782	036244	004737	037026		UU11:	JSR	PC,UUSUB		
4783	036250	040200	000000	000000		.WORD	40200,0,0,0		:AC
4784	036260	040177	072727	127272		.WORD	40177,72727,127272,72727		:FSRC
4785	036270	040177	072727	127272		.WORD	40177,72727,127272,72727		:FRACTIONAL RES.
4786	036300	000000	000000	000000		.WORD	0,0,0,0		:INTEGER RES.
4787	036310	000217	000200			.WORD	217,200		:FPS'S
4788									
4789									
4790									
4791	036314	004737	037026		UU12:	JSR	PC,UUSUB		
4792	036320	003377	177777	177777		.WORD	3377,-1,-1,52525		:AC

4793 036330 040200 000000 000000
4794 036340 003377 177777 177777
4795 036350 000000 000000 000000
4796 036360 000217 000200
4797
4798
4799
4800 036364 004737 037026
4801 036370 040300 000000 000000
4802 036400 020200 000000 000000
4803 036410 020300 000000 000000
4804 036420 000000 000000 000000
4805 036430 000217 000200
4806
4807
4808
4809 036434 004737 037016
4810 036440 060252 125252 125252
4811 036450 060300 000000 000000
4812 036460 000000 000000 000000
4813 036470 000377 177777 177777
4814 036500 041217 141206
4815
4816
4817
4818 036504 004737 037016
4819 036510 060252 125252 125252
4820 036520 060100 000000 000000
4821 036530 000000 000000 000000
4822 036540 000177 177777 177777
4823 036550 041217 141206
4824
4825 036554 004737 037016
4826 036560 060252 125252 125252
4827 036570 060100 000000 000000
4828 036600 000000 000000 000000
4829 036610 000000 000000 000000
4830 036620 000217 000206
4831
4832
4833
4834 036624 004737 037016
4835 036630 020252 125252 125252
4836 036640 017700 000000 000000
4837 036650 077777 177777 177777
4838 036660 000000 000000 000000
4839 036670 042217 142200
4840
4841
4842
4843 036674 004737 037016
4844 036700 020252 125252 125252
4845 036710 020100 000000 000000
4846 036720 000177 177777 177777
4847 036730 000000 000000 000000
4848 036740 042217 142204

.WORD 40200,0,0,0 ; FSRC
.WORD 3377,-1,-1,52525 ; FRACTIONAL RES.
.WORD 0,0,0,0 ; INTEGER RES.
.WORD 217,200 ; FPS'S
: MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.
UU13: JSR PC,UUSUB
.WORD 40300,0,0,0 ; AC
.WORD 20200,0,0,1 ; FSRC
.WORD 20300,0,0,2 ; FRACTIONAL RES.
.WORD 0,0,0,0 ; INTEGER RES.
.WORD 217,200 ; FPS'S
: OVERFLOW -- FINAL EXP > 128.
UU14: JSR PC,UUSUB1
.WORD 60252,125252,125252,125252 ; AC
.WORD 60300,0,0,0 ; FSRC
.WORD 0,0,0,0 ; FRAC...
.WORD 377,-1,-1,-2 ; ...AND INT RESULT (FIV - 1).
.WORD 41217,141206 ; FPS'S
: OVERFLOW -- FINAL EXP = 128.
UU15: JSR PC,UUSUB1
.WORD 60252,125252,125252,125252 ; AC
.WORD 60100,0,0,0 ; FSRC
.WORD 0,0,0,0 ; FRAC...
.WORD 177,-1,-1,-1 ; ...AND INT RESULT (FIV = 1).
.WORD 41217,141206 ; FPS'S
: SAME OPERANDS, FIV = 0.
JSR PC,UUSUB1
.WORD 60252,125252,125252,125252
.WORD 60100,0,0,0
.WORD 0,0,0,0 ; FRAC...
.WORD 0,0,0,0 ; ...AND INT RESULT (FIV - 0).
.WORD 217,206
: UNDERFLOW -- FINAL EXP < -128.
UU16: JSR PC,UUSUB1
.WORD 20252,125252,125252,125252 ; AC
.WORD 17700,0,0,0 ; FSRC
.WORD 077777,-1,-1,-1 ; FRAC...
.WORD 0,0,0,0 ; ...AND INT RESULT (FIU = 1).
.WORD 42217,142200 ; FPS'S
: UNDERFLOW -- FINAL EXP = -128.
UU17: JSR PC,UUSUB1
.WORD 20252,125252,125252,125252 ; AC
.WORD 20100,0,0,0 ; FSRC
.WORD 000177,-1,-1,-1 ; FRAC...
.WORD 0,0,0,0 ; ...AND INT RESULT (FIU = 1).
.WORD 42217,142204 ; FPS'S

```

4849
4850 036744 004737 037016 JSR PC,UUSUB1 ; SAME OPERANDS, FIU = 0.
4851 036750 020252 125252 .WORD 20252,125252,125252,125252
4852 036760 020100 000000 000000 .WORD 20100,0,0,0
4853 036770 000000 000000 000000 .WORD 0,0,0,0 ; FRAC...
4854 037000 000000 000000 000000 .WORD 0,0,0,0 ;...AND INT RESULT (FIU = 0).
4855 037010 000217 000204 .WORD 217,204
4856
4857 037014 000525 BR UUDONE ; FINISHED.
4858
4859 : SUBROUTINE TO SET-UP, EXECUTE, AND CHECK THE RESULTS OF MODD.
4860
4861 : CALL:
4862 : JSR PC,UUSUB ; R UUSUB1
4863 : .WORD X,X,X,X ; A OPERAND (MULTIPLICAND).
4864 : .WORD X,X,X,X ; FSRC OPERAND (MULTIPLIER).
4865 : .WORD X,X,X,X ; RESULTING FRACTION (AC0).
4866 : .WORD X,X,X,X ; RESULTING INTEGER (AC1).
4867 : .WORD X,X ; FPS BEFORE AND EXP FPS AFTER.
4868 : ; RETURN TO CALLER+44
4869
4870 : THE OPERANDS ARE SELECTED USING THE CALLERS PC.
4871 : AFTER EXECUTION THE RESULTING INTEGER, FRACTION, AND FPS
4872 : ARE TESTED. IF ALL OK, RETURN.
4873 : OTHERWISE, REPORT ERROR AND RETURN.
4874
4875 037016 112737 000040 056304 UUSUB1: MOVB #40,EM221X ; ENABLE MESSAGE EXTENSION.
4876 037024 000402 SKP2
4877 037026 105037 056304 UUSUB: CLRB EM221X ; DISABLE MESSAGE EXTENSION.
4878 037032 012637 001164 MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC...
4879 037036 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
4880 037040 013701 001164 MOV $TMP2,R1 ;...AND USE AS ARG POINTER.
4881 037044 170011 SETD
4882 037046 172427 000000 LDD #0,AC0 ; CLEAR AC0.
4883 037052 172527 000000 LDD #0,AC1 ; AND AC1.
4884 037056 016100 000040 MOV 40(R1),R0
4885 037062 170100 LDFPS R0 ; SET INITIAL FPS.
4886 037064 010100 MOV R1,R0
4887 037066 172410 LDD (R0),AC0 ; LOAD THE MULTIPLICAND.
4888 037070 010037 001172 MOV R0,$TMP5
4889 037074 010100 MOV R1,R0
4890 037076 062700 000010 ADD #10,R0 ;POINT TO MULTIPLIER.
4891 037102 010037 001174 MOV R0,$TMP6
4892 037106 171410 2$: MODD (R0),AC0 ;TEST INSTRUCTION.
4893 037110 000240 NOP
4894 037112 170204 STFPS R4 ;GET THE FPS.
4895 037114 010437 001166 MOV R4,$TMP3
4896 037120 016137 000042 001170 MOV 42(R1),$TMP4 ; EXP FPS.
4897 037126 170011 SETD
4898 037130 012700 037250 MOV #20$,R0
(1) 037134 174010 STD AC0,(R0) ;: STORE AC0
4899 037136 010037 001200 MOV R0,$TMP10 ;...AND SET POINTER.
4900 037142 012700 037260 MOV #21$,R0
(1) 037146 174110 STD AC1,(R0) ;: STORE AC1
4901 037150 010037 001204 MOV R0,$TMP12 ;...AND SET POINTER.
4902 037154 010100 MOV R1,R0

```

4903	037156	062700	000020			ADD	#20,R0	
4904	037162	010037	001176			MOV	R0,\$TMP7	; POINTS TO EXP FRACTION.
4905	037166	010037	037212			MOV	R0,3\$; IN THE CHECKER TOO.
4906	037172	062700	000010			ADD	#10,R0	
4907	037176	010037	001202			MOV	R0,\$TMP11	;...AND EXP INTEGER.
4908	037202	010037	037224			MOV	R0,4\$	
4909	037206	004537	004312			JSR	R5,CHECK4	; FRACTION CORRECT ??
4910	037212	000000	037250	3\$:		O,20\$		
4911	037216	001011				BNE	5\$; BR IF NOT.
4912	037220	004537	004312			JSR	R5,CHECK4	; YES, INTEGER CORRECT ??
4913	037224	000000	037260	4\$:		O,21\$		
4914	037230	001004				BNE	5\$; BR IF NOT.
4915	037232	023737	001166	001170		CMP	\$TMP3,\$TMP4	; FPS RIGHT ??
4916	037240	001401				BEQ	6\$; RETURN IF ALL OK.
4917								
4918	037242	104221			5\$:	ERROR	221	; FPS WRONG.
4919								
4920	037244	000161	000044		6\$:	JMP	44(R1)	; RETURN
4921								
4922	037250	000000	000000	000000	20\$:	.WORD	0,0,0,0	; FRACTION BUFFER.
4923	037260	000000	000000	000000	21\$:	.WORD	0,0,0,0	; INTEGER BUFFER.
4924								
4925	037270					UUDONE:		
(1)	037270	104412				CLRFPS		:: CLEAR FP STATUS...
(2)	037272	000400				BR	\$EOP	::...AND SIGNAL END-PASS
4926								
4927		000051				LASTST-	\$TN-1	; REMEMBER LAST TEST NUMBER.


```

037702 032777 004000 141230 38. BIT #BIT11,@SWR ::INHIBIT ITERATIONS?
037710 001011 BNE 1$ ::BR IF YES
037712 005737 001242 TST $PASS ::IF FIRST PASS OF PROGRAM
037716 001406 BEQ 1$ :: INHIBIT ITERATIONS
037720 005237 001104 INC $ICNT ::INCREMENT ITERATION COUNT
037724 023737 001220 001104 CMP $TIMES,$ICNT ::CHECK THE NUMBER OF ITERATIONS MADE
037732 002024 BGE $OVER ::BR IF MORE ITERATION REQUIRED
037734 012737 000001 001104 1$: MOV #1,$ICNT ::REINITIALIZE THE ITERATION COUNTER
037742 013737 040020 001220 MOV $MXCNT,$TIMES ::SET NUMBER OF ITERATIONS TO DO
037750 105237 001102 $SVLAD: INCB $STNM ::COUNT TEST NUMBERS
037754 113737 001102 001240 MOV $STNM,$TESTN ::SET TEST NUMBER IN APT MAILBOX
037762 011637 001106 MOV (SP),$LPADR ::SAVE SCOPE LOOP ADDRESS
037766 011637 001110 MOV (SP),$LPERR ::SAVE ERROR LOOP ADDRESS
037772 005037 001222 CLR $ESCAPE ::CLEAR THE ESCAPE FROM ERROR ADDRESS
037776 112737 000001 001115 MOV #1,$ERMAX ::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
040004 013777 001102 141130 $OVER: MOV $STNM,@DISPLAY ::DISPLAY TEST NUMBER
040012 013716 001106 MOV $LPADR,(^P) ::FUDGE RETURN ADDRESS
040016 000452 BR SCOPEX
040020 000001 $MXCNT: 1 ::MAX. NUMBER OF ITERATIONS
040022 $SW08TBL:
(3) 040022 004360 .WORD TST1+2 ::STARTING ADDRESS OF TEST 1
(3) 040024 004542 .WORD TST2+2 ::STARTING ADDRESS OF TEST 2
(3) 040026 004664 .WORD TST3+2 ::STARTING ADDRESS OF TEST 3
(3) 040030 005136 .WORD TST4+2 ::STARTING ADDRESS OF TEST 4
(3) 040032 005324 .WORD TST5+2 ::STARTING ADDRESS OF TEST 5
(3) 040034 005464 .WORD TST6+2 ::STARTING ADDRESS OF TEST 6
(3) 040036 006476 .WORD TST7+2 ::STARTING ADDRESS OF TEST 7
(3) 040040 007300 .WORD TST10+2 ::STARTING ADDRESS OF TEST 10
(3) 040042 010052 .WORD TST11+2 ::STARTING ADDRESS OF TEST 11
(3) 040044 010574 .WORD TST12+2 ::STARTING ADDRESS OF TEST 12
(3) 040046 011420 .WORD TST13+2 ::STARTING ADDRESS OF TEST 13
(3) 040050 011764 .WORD TST14+2 ::STARTING ADDRESS OF TEST 14
(3) 040052 012222 .WORD TST15+2 ::STARTING ADDRESS OF TEST 15
(3) 040054 012456 .WORD TST16+2 ::STARTING ADDRESS OF TEST 16
(3) 040056 012716 .WORD TST17+2 ::STARTING ADDRESS OF TEST 17
(3) 040060 013160 .WORD TST20+2 ::STARTING ADDRESS OF TEST 20
(3) 040062 013606 .WORD TST21+2 ::STARTING ADDRESS OF TEST 21
(3) 040064 014234 .WORD TST22+2 ::STARTING ADDRESS OF TEST 22
(3) 040066 014622 .WORD TST23+2 ::STARTING ADDRESS OF TEST 23
(3) 040070 015224 .WORD TST24+2 ::STARTING ADDRESS OF TEST 24
(3) 040072 015626 .WORD TST25+2 ::STARTING ADDRESS OF TEST 25
(3) 040074 016214 .WORD TST26+2 ::STARTING ADDRESS OF TEST 26
(3) 040076 016624 .WORD TST27+2 ::STARTING ADDRESS OF TEST 27
(3) 040100 017070 .WORD TST30+2 ::STARTING ADDRESS OF TEST 30
(3) 040102 017326 .WORD TST31+2 ::STARTING ADDRESS OF TEST 31
(3) 040104 020064 .WORD TST32+2 ::STARTING ADDRESS OF TEST 32
(3) 040106 021102 .WORD TST33+2 ::STARTING ADDRESS OF TEST 33
(3) 040110 022330 .WORD TST34+2 ::STARTING ADDRESS OF TEST 34
(3) 040112 023342 .WORD TST35+2 ::STARTING ADDRESS OF TEST 35
(3) 040114 024040 .WORD TST36+2 ::STARTING ADDRESS OF TEST 36
(3) 040116 024576 .WORD TST37+2 ::STARTING ADDRESS OF TEST 37
(3) 040120 025624 .WORD TST40+2 ::STARTING ADDRESS OF TEST 40
(3) 040122 026646 .WORD TST41+2 ::STARTING ADDRESS OF TEST 41
(3) 040124 027634 .WORD TST42+2 ::STARTING ADDRESS OF TEST 42
(3) 040126 030514 .WORD TST43+2 ::STARTING ADDRESS OF TEST 43
(3) 040130 031172 .WORD TST44+2 ::STARTING ADDRESS OF TEST 44

```



```

(1) 040422 000770          BR      7$          ;;LOOP
(1)
(1)                      ;HORIZONTAL TAB PROCESSOR
(1)
(1) 040424 112716 000040 8$:      MOVB   #' , (SP)      ;;REPLACE TAB WITH SPACE
(1) 040430 004737 040450 9$:      JSR     PC, $TYPEC      ;;TYPE A SPACE
(1) 040434 132737 000007 040566 BITB   #7, $CHARCNT      ;;BRANCH IF NOT AT
(1) 040442 001372          BNE     9$          ;;TAB STOP
(1) 040444 005726          TST    (SP)+          ;;POP SPACE OFF STACK
(1) 040446 000724          BR      2$          ;;GET NEXT CHARACTER
(1) 040450          $TYPEC:
(1) 040450 105777 140470 TSTB   @STKS          ;;CHAR IN KYBD BUFFER?      ;MJD001
(1) 040454 100022          BPL    10$          ;;BR IF NOT                  ;MJD001
(1) 040456 017746 140464 MOV     @STKB, -(SP)    ;;GET CHAR                    ;MJD001
(1) 040462 042716 177600 BIC    #177600, (SP)   ;;STRIP EXTRANEOUS BITS      ;MJD001
(1) 040466 122716 000023 CMPB   #$XOFF, (SP)    ;;WAS CHAR XOFF              ;MJD001
(1) 040472 001012          BNE    102$         ;;BR IF NOT                  ;MJD001
(1) 040474          101$:
(1) 040474 105777 140444 TSTB   @STKS          ;;WAIT FOR CHAR              ;MJD001
(1) 040500 100375          BPL    101$         ;;MJD001
(1) 040502 117716 140440 MOVB   @STKB, (SP)     ;;GET CHAR                    ;MJD001
(1) 040506 042716 177600 BIC    #177600, (SP)   ;;STRIP IT                    ;MJD001
(1) 040512 122716 000021 CMPB   #$XON, (SP)     ;;WAS IT XON?                ;MJD001
(1) 040516 001366          BNE    101$         ;;BR IF NOT                  ;MJD001
(1) 040520          102$:
(1) 040520 005726          TST    (SP)+          ;;FIX STACK                  ;MJD001
(1) 040522          10$:
(1) 040522 105777 140422 TSTB   @STPS          ;;WAIT UNTIL PRINTER IS READY ;MJD001
(1) 040526 100375          BPL    10$          ;;MJD001
(1) 040530 116677 000002 140414 MOVB   2(SP), @STPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 040536 122766 000015 000002 CMPB   #CR, 2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
(1) 040544 001003          BNE    1$          ;;BRANCH IF NO
(1) 040546 105037 040566 CLRB   $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
(1) 040552 000406          BR     $TYPEX      ;;EXIT
(1) 040554 122766 000012 000002 1$:      CMPB   #LF, 2(SP)    ;;IS CHARACTER A LINE FEED?
(1) 040562 001402          BEQ    $TYPEX      ;;BRANCH IF YES
(1) 040564 105227          INCB   (PC)+          ;;COUNT THE CHARACTER
(1) 040566 000000          $CHARCNT: .WORD 0    ;;CHARACTER COUNT STORAGE
(1) 040570 000207          $TYPEX: RTS      PC
(1)
4963 ;////////////////////////////////////
4964 ; NOW BACK UP AND PATCH $TYPEC TO ENABLE <^G> RECOGNITION
4965 ; DURING OUTPUT SEQUENCES.
4966 ;
4967          040572          SVPC=          ; SAVE THE PC...
4968          040520          --          $TYPEC+50      ; ...AND POINT TO 102$.
4969 040520 000424 102$:      BR      $$TCX          ; BR TO PATCH...
4970 040522          10$:          --          SVPC          ; ...AND RETURN.
4971          040572          ; RESTORE PC.
4972
4973 040572 122627 000007 $$TCX:      CMPB   (SP)+, #7      ; <^G> ???
4974 040576 001020          BNE    1$          ; SKIP IF NOT.
4975          .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
(1) 040600 005737 000042 TST    @#42          ;;ARE WE RUNNING UNDER XXDP/ACT?
(1) 040604 001012          BNE    64$          ;;BRANCH IF YES
(1) 040606 123727 001254 000001 CMPB   $ENV, #1      ;;ARE WE RUNNING UNDER APT?
    
```

```

(1) 040614 001406          BEQ    64$      ;;BRANCH IF YES
(1) 040616 023727 001140 000176  CMP    SWR,#SWREG  ;;SOFTWARE SWITCH REG SELECTED?
(1) 040624 001005          BNE    65$      ;;BRANCH IF NO
(1) 040626 104406          GTSWR                ;;GET SOFT-SWR SETTINGS
(1) 040630 000403          BR     65$
(1) 040632 112737 000001 001134 64$:  MOVB   #1,$AUTOB  ;;SET AUTO-MODE INDICATOR
(1) 040640 000730          65$:
4976 040640 000730          1$:  BR     $TYPEC+52  ; RETURN.
4977                               ;////////////////////

```


(1)	041004	005204		4\$:	INC	R4	::DON'T SUPPRESS ANYMORE 0'S
(1)	041006	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
(1)	041012	052703	000040	5\$:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY
(1)	041016	110337	041062		MOVB	R3,8\$::SAVE FOR TYPING
(1)	041022	104401	041062		TYPE	,8\$::GO TYPE THIS DIGIT
(1)	041026	105337	041064	7\$:	DECB	\$OCNT	::COUNT BY 1
(1)	041032	003347			BGT	2\$::BR IF MORE TO DO
(1)	041034	002402			BLT	6\$::BR IF DONE
(1)	041036	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
(1)	041040	000744			BR	2\$::GO DO THE LAST DIGIT
(1)	041042	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
(1)	041044	012604			MOV	(SP)+,R4	::RESTORE R4
(1)	041046	012603			MOV	(SP)+,R3	::RESTORE R3
(1)	041050	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
(1)	041056	012616			MOV	(SP)+,(SP)	
(1)	041060	000002			RTI		::RETURN
(1)	041062	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
(1)	041063	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
(1)	041064	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
(1)	041065	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
(1)	041066	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE


```

4993 .SBTTL APT COMMUNICATIONS ROUTINE
(1)
(2)
(1) 041314 112737 000001 041560 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
(1) 041322 112737 000001 041556 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
(1) 041330 000403 BR $ATYC
(1) 041332 112737 000001 041560 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
(1) 041340 $ATYC:
(3) 041340 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 041342 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(1) 041344 105737 041556 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
(1) 041350 001450 BEQ 5$ ;;IF NOT: BR
(1) 041352 122737 000001 001254 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
(1) 041360 001031 BNE 3$ ;;IF NOT: BR
(1) 041362 132737 000100 001255 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 041370 001425 BEQ 3$ ;;IF NOT: BR
(1) 041372 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
(1) 041376 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 041404 005737 001234 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
(1) 041410 001375 BNE 1$ ;;IF NOT: WAIT
(1) 041412 010037 001250 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
(1) 041416 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 041420 001376 BNE 2$
(1) 041422 163700 001250 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 041426 006200 ASR R0 ;;GET MESSAGE LNGTH IN WORDS
(1) 041430 010037 001252 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
(1) 041434 012737 000004 001234 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
(1) 041442 000413 BR 5$
(1) 041444 017637 000004 041470 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
(1) 041452 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
(3) 041460 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
(1) 041464 004737 040236 JSR PC,$TYPE ;;CALL TYPE MACRO
(1) 041470 000000 4$: .WORD 0
(1) 041472 5$:
(1) 041472 105737 041560 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
(1) 041476 001416 BEQ 12$ ;;IF NOT: BR
(1) 041500 005737 001254 TST $ENV ;;RUNNING UNDER APT?
(1) 041504 001413 BEQ 12$ ;;IF NOT: BR
(1) 041506 005737 001234 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
(1) 041512 001375 BNE 11$ ;;IF NOT: WAIT
(1) 041514 017637 000004 001236 MOV @4(SP),$FATAL ;;GET ERROR #
(1) 041522 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 041530 005237 001234 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
(1) 041534 105037 041560 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
(1) 041540 105037 041557 CLRB $LFLG ;;CLEAR LOG FLAG
(1) 041544 105037 041556 CLRB $MFLG ;;CLEAR MESSAGE FLAG
(3) 041550 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
(3) 041552 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
(1) 041554 000207 RTS PC ;;RETURN
(1) 041556 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
(1) 041557 000 $LFLG: .BYTE 0 ;;LOG FLAG
(1) 041560 000 $FFLG: .BYTE 0 ;;FATAL FLAG
(1) 041562 .EVEN
(1) 000200 APTSIZE-200
(1) 000001 APTENV-001
(1) 000100 APTSPOOL-100
  
```

(1) 000040

APTCSUP=040

```

4995      .SBITL  TTY INPUT ROUTINE
(1)
(2)      ::*****
(1)      .ENABL  LSB
(1)
(2)      ::*****
(1)      ::*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)      ::*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1)      ::*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1)      ::*WHEN OPERATING IN TTY FLAG MODE.
(1) 041562 022737 000176 001140 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED?
(1) 041570 001074                BNE      15$          ;; BRANCH IF NO
(1) 041572 105777 137346        TSTB     @STKS        ;; CHAR THERE?
(1) 041576 100071                BPL      15$          ;; IF NO, DON'T WAIT AROUND
(1) 041600 117746 137342        MOVB     @STKB,-(SP)  ;; SAVE THE CHAR
(1) 041604 042716 177600        BIC      #^C177,(SP) ;; STRIP-OFF THE ASCII
(1) 041610 022726 000007        CMP      #7,(SP)+    ;; IS IT A CONTROL G?
(1) 041614 001062                BNE      15$          ;; NO, RETURN TO USER
(1) 041616 123727 001134 000001  CMPB     $AUTOB,#1   ;; ARE WE RUNNING IN AUTO-MODE?
(1) 041624 001456                BEQ      15$          ;; BRANCH IF YES
(1)
(1) 041626 104401 042171                TYPE     ,SCNTLG     ;; ECHO THE CONTROL-G (^G)
(1) 041632 104401 042176        $GTSWR: TYPE     ,SMSWR     ;; TYPE CURRENT CONTENTS
(2) 041636 013746 000176        MOV      SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
(2) 041642 104402                TYPOC    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 041644 104401 042207                TYPE     ,SMNEW     ;; PROMPT FOR NEW SWR
(1) 041650 005046        19$:  CLR      -(SP)   ;; CLEAR COUNTER
(1) 041652 005046                CLR      -(SP)     ;; THE NEW SWR
(1) 041654 105777 137264        7$:  TSTB     @STKS    ;; CHAR THERE?
(1) 041660 100375                BPL      7$         ;; IF NOT TRY AGAIN
(1)
(1) 041662 117746 137260                MOVB     @STKB,-(SP) ;; PICK UP CHAR
(1) 041666 042716 177600                BIC      #^C177,(SP) ;; MAKE IT 7-BIT ASCII
(1)
(1)
(1) 041672 021627 000025        9$:  CMP      (SP),#25  ;; IS IT A CONTROL-U?
(1) 041676 001005                BNE      10$        ;; BRANCH IF NOT
(1) 041700 104401 042164                TYPE     ,SCNTLU    ;; YES, ECHO CONTROL-U (^U)
(1) 041704 062706 000006        20$: ADD      #6,SP     ;; IGNORE PREVIOUS INPUT
(1) 041710 000757                BR       19$        ;; LET'S TRY IT AGAIN
(1)
(1)
(1) 041712 021627 000015        10$: CMP      (SP),#15   ;; IS IT A <CR>?
(1) 041716 001022                BNE      16$        ;; BRANCH IF NO
(1) 041720 005766 000004                TST      4(SP)      ;; YES, IS IT THE FIRST CHAR?
(1) 041724 001403                BEQ      11$        ;; BRANCH IF YES
(1) 041726 016677 000002 137204        MOV      2(SP),@SWR  ;; SAVE NEW SWR
(1) 041734 062706 000006        11$: ADD      #6,SP     ;; CLEAR UP STACK
(1) 041740 104401 001231        14$: TYPE     ,SCRLF   ;; ECHO <CR> AND <LF>
(1) 041744 123727 001135 000001        CMPB     $INTAG,#1  ;; RE-ENABLE TTY KBD INTERRUPTS?
(1) 041752 001003                BNE      15$        ;; BRANCH IF NOT
(1) 041754 012777 000100 137162        MOV      #100,@STKS ;; RE-ENABLE TTY KBD INTERRUPTS
(1) 041762 000002                RTI                     ;; RETURN
(1) 041764 004737 040450        15$: JSR      PC,$TYPEC  ;; ECHO CHAR
(1) 041770 021627 000060        16$: CMP      (SP),#60  ;; CHAR < 0?
    
```

```

(1) 041774 002420          BLT      18$          ;;BRANCH IF YES
(1) 041776 021627 000067  CMP      (SP),#67      ;;CHAR > 7?
(1) 042002 003015          BGT      18$          ;;BRANCH IF YES
(1) 042004 042726 000060  BIC      #60,(SP)+     ;;STRIP-OFF ASCII
(1) 042010 005766 000002  TST      2(SP)         ;;IS THIS THE FIRST CHAR
(1) 042014 001403          BEQ      17$          ;;BRANCH IF YES
(1) 042016 006316          ASL      (SP)         ;;NO, SHIFT PRESENT
(1) 042020 006316          ASL      (SP)         ;;CHAR OVER TO MAKE
(1) 042022 006316          ASL      (SP)         ;;ROOM FOR NEW ONE.
(1) 042024 005266 000002  17$: INC      2(SP)         ;;KEEP COUNT OF CHAR
(1) 042030 056616 177776  BIS      -2(SP),(SP)   ;;SET IN NEW CHAR
(1) 042034 000707          BR       7$           ;;GET THE NEXT ONE
(1) 042036 104401 001230  18$: TYPE   $QUES      ;;TYPE ?<CR><LF>
(1) 042042 000720          BR      20$          ;;SIMULATE CONTROL-U
(1) .DSABL  LSB
(1)
(1)
(2)
(1) *****
(1) *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1) *CALL:
(1) * RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1) * RETURN HERE   ;;CHARACTER IS ON THE STACK
(1) *              ;;WITH PARITY BIT STRIPPED OFF
(1)
(1) $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
(1) 042044 011646          MOV      4(SP),2(SP)   ;;SAVE THE PS
(1) 042046 016666 000004 000002  1$: TST      @STKS      ;;WAIT FOR
(1) 042054 105777 137064          BPL      1$           ;;A CHARACTER
(1) 042060 100375          MOVB     @STKB,4(SP)   ;;READ THE TTY
(1) 042062 117766 137060 000004          BIC      #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
(1) 042070 042766 177600 000004          CMP      4(SP),#23    ;;IS IT A CONTROL-S?
(1) 042076 026627 000004 000023          BNE      3$           ;;BRANCH IF NO
(1) 042104 001013          TST      @STKS      ;;WAIT FOR A CHARACTER
(1) 042106 105777 137032          BPL      2$           ;;LOOP UNTIL ITS THERE
(1) 042112 100375          MOVB     @STKB,-(SP)  ;;GET CHARACTER
(1) 042114 117746 137026          BIC      #^C<177>,(SP) ;;MAKE IT 7-BIT ASCII
(1) 042120 042716 177600          CMP      (SP)+,#21    ;;IS IT A CONTROL-Q?
(1) 042124 022627 000021          BNE      2$           ;;IF NOT DISCARD IT
(1) 042130 001366          BR       1$           ;;YES, RESUME
(1) 042132 000750          CMP      4(SP),#140   ;;IS IT UPPER CASE?
(1) 042134 026627 000004 000140          BLT      4$           ;;BRANCH IF YES
(1) 042142 002407          CMP      4(SP),#175   ;;IS IT A SPECIAL CHAR?
(1) 042144 026627 000004 000175          BGT      4$           ;;BRANCH IF YES
(1) 042152 003003          BIC      #40,4(SP)    ;;MAKE IT UPPER CASE
(1) 042154 042766 000040 000004          RTI          ;;GO BACK TO USER
(1) 042162 000002          $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
(1) 042164 052536 005015 000          $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
(1) 042171 136 006507 000012          $MSWR: .ASCIZ <15><12>/SWR = /
(1) 042176 005015 053523 020122          $MNEW: .ASCIZ / NEW - /
(1) 042207 040 047040 053505
  
```

4997
 (1)
 (2)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1) 042220 010046
 (1) 042222 016600 000002
 (1) 042226 005740
 (1) 042230 111000
 (1) 042232 006300
 (1) 042234 016000 042254
 (1) 042240 000200
 (1)
 (1)
 (1)
 (1) 042242 011646
 (1) 042244 016666 000004 000002
 (1) 042252 000002
 (1)
 (3)
 (3)
 (3)
 (3)
 (3)
 (3)
 (3) 042254 042242
 (3) 042256 040236
 (3) 042260 040666
 (3) 042262 040642
 (3) 042264 040702
 (3) 042266 041070
 (1)
 (3) 042270 041632
 (1)
 (3) 042272 041562
 (3) 042274 042044
 4998 042276 042302
 4999 042300 042310
 5000 000026
 5001
 5002
 5003
 5004
 5005 042302 011637 001110
 5006 042306 000002
 5007
 5008
 5009
 5010
 5011 042310 012700 042342
 5012 042314 010037 000244

.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.

```
$TRAP:  MOV    RO,-(SP)           ;;SAVE R0
        MOV    2(SP),RO         ;;GET TRAP ADDRESS
        TST    -(RO)           ;;BACKUP BY 2
        MOVB   (RO),RO         ;;GET RIGHT BYTE OF TRAP
        ASL    RO              ;;POSITION FOR INDEXING
        MOV    $TRPAD(RO),RO   ;;INDEX TO TABLE
        RTS    RO              ;;GO TO ROUTINE
```

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

```
$TRAP2: MOV    (SP),-(SP)       ;;MOVE THE PC DOWN
        MOV    4(SP),2(SP)     ;;MOVE THE PSW DOWN
        RTI                    ;;RESTORE THE PSW
```

.SBTTL TRAP TABLE

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

```
ROUTINE
-----
$TRPAD: .WORD  $TRAP2           TRAP+1(104401)  TTY TYPEOUT ROUTINE
        .TYPE  ;;CALL=TYPE     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        .TYPOC ;;CALL=TYPOC   TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        .TYPOS ;;CALL=TYPOS   TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        .TYPON ;;CALL=TYPON   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
        .TYPDS ;;CALL=TYPDS
        $GTSWR ;;CALL=GTSWR   TRAP+6(104406)  GET SOFT-SWR SETTING
        $CKSWR ;;CALL=CKSWR   TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR   TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        .LPER  ;;CALL=LUPERR  TRAP+11(104411) SET LOOP-ON-ERROR ADDRESS
        .CLRFP ;;CALL=CLRFP   TRAP+12(104412) CLEAR FPS AND END TEST
```

\$TERM=-\$TRPAD

;; THIS IS TO SET THE LOOP-ON-ERROR (SWR9) ADDRESS.
 ;; CALLED VIA 'TRAP+11'.

```
.LPER:  MOV    (SP),$LPERR
        RTI
```

;; COME HERE AT END OF EACH TEST TO CLEAR FPS AND RESET VECTORS.
 ;; CALLED VIA 'TRAP+12'.

```
.CLRFP: MOV    #1$,RO
        MOV    RO,FPVEC      ; JUST IN CASE THE...
```

5013	042320	010037	000010			MOV	R0,RESVEC		;...LDFPS IS BAD.
5014	042324	010037	000004			MOV	R0,ERRVEC		
5015	042330	005000				CLR	R0		
5016	042332	170100				LDFPS	R0		; CLEAR FP STATUS.
5017	042334	000240	000240			240,240			
5018	042340	000401				SKP1			
5019	042342	022626		1\$:		CMP	(SP)+,(SP)+		; IF IT TRAPPED, JUST FIX THE STACK.
5020	042344	012737	004200	000244		MOV	#TRP244,FPVEC		; RESET ALL VECTORS.
5021	042352	012737	004234	000010		MOV	#TRP10,RESVEC		
5022	042360	012737	004224	000004		MOV	#TRP04,ERRVEC		
5023	042366	000002				RTI			

```

5025      .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)
(1)
(1) 042370 012737 042534 000024 $PWRDN: MOV  # $ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1) 042376 012737 000340 000026      MOV  #340,@#PWRVEC+2 ;;PRIO:7
(3) 042404 010046      MOV  R0,.(SP) ;;PUSH R0 ON STACK
(3) 042406 010146      MOV  R1,-(SP) ;;PUSH R1 ON STACK
(3) 042410 010246      MOV  R2,-(SP) ;;PUSH R2 ON STACK
(3) 042412 010346      MOV  R3,-(SP) ;;PUSH R3 ON STACK
(3) 042414 010446      MOV  R4,-(SP) ;;PUSH R4 ON STACK
(3) 042416 010546      MOV  R5,-(SP) ;;PUSH R5 ON STACK
(3) 042420 017746 136514      MOV  @SWR,-(SP) ;;PUSH @SWR ON STACK
(1) 042424 010637 042540      MOV  SP,$SAVR6 ;;SAVE SP
(1) 042430 012737 042442 000024      MOV  # $PWRUP,@#PWRVEC ;;SET UP VECTOR
(1) 042436 000000      HALT
(1) 042440 000776      BR   .-2 ;;HANG UP
(1)
(2)
(1)
(1) 042442 012737 042534 000024 $PWRUP: MOV  # $ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1) 042450 013706 042540      MOV  $SAVR6,SP ;;GET SP
(1) 042454 005037 042540      CLR  $SAVR6 ;;WAIT LOOP FOR THE TTY
(1) 042460 005237 042540 1$: INC  $SAVR6 ;;WAIT FOR THE INC
(1) 042464 001375      BNE  1$ ;;OF WORD
(3) 042466 012677 136446      MOV  (SP)+,@SWR ;;POP STACK INTO @SWR
(3) 042472 012605      MOV  (SP)+,R5 ;;POP STACK INTO R5
(3) 042474 012604      MOV  (SP)+,R4 ;;POP STACK INTO R4
(3) 042476 012603      MOV  (SP)+,R3 ;;POP STACK INTO R3
(3) 042500 012602      MOV  (SP)+,R2 ;;POP STACK INTO R2
(3) 042502 012601      MOV  (SP)+,R1 ;;POP STACK INTO R1
(3) 042504 012600      MOV  (SP)+,R0 ;;POP STACK INTO R0
(1) 042506 012737 042370 000024      MOV  # $PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 042514 012737 000340 000026      MOV  #340,@#PWRVEC+2 ;;PRIO:7
(1) 042522 104401      TYPE REPORT THE POWER FAILURE
(1) 042524 043304 $PWRMG: .WORD POWERM ;;POWER FAIL MESSAGE POINTER
(1) 042526 012716      MOV  (PC)+,(SP) ;;RESTART AT START
(1) 042530 003542 $PWRAD: .WORD START ;;RESTART ADDRESS
(1) 042532 000002      RTI
(1) 042534 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
(1) 042536 000776      BR   .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
(1) 042540 000000      $SAVR6: 0 ;;PUT THE SP HERE
    
```



```

5029          .SBTTL  ERROR TYPE OUT ROUTINE
5030
5031          : THIS ROUTINE IS CALLED FROM $ERROR, TO PRINT THE VARIOUS
5032          : ERROR SIGNATURES.
5033
5034 042742 010046          ERTYPE: MOV      R0,-(SP)          ;SAVE REGISTERS.
5035 042744 010146          MOV      R1,-(SP)
5036 042746 010246          MOV      R2,-(SP)
5037 042750 010346          MOV      R3,-(SP)
5038 042752 104401 001231          TYPE      , $CRLF
5039 042756 113737 001102 001160          MOVB     $TSTNM,$TMP0
5040 042764 042737 177400 001160          BIC      #^C377,$TMP0          ; SET TEST NUMBER...
5041 042772 013737 001116 001162          MOV      $ERRPC,$TMP1          ; ...AND CALL PC.
5042 043000 113700 001114          MOVB     $ITEMB,R0          ; GET ERROR ITEM NUMBER.
5043 043004 042700 177400          BIC      #^C377,R0
5044 043010 001524          BEQ      ERT4          ; JUST IGNORE ERROR 0.
5045
5046 043012 005300          2$:    DEC      R0          ; NOW MAKE INTO AN INDEX.
5047 043014 006300          ASL      R0
5048 043016 006300          ASL      R0
5049 043020 006300          ASL      R0
5050 043022 062700 001312          ADD      #$ERRTB,R0          ; R0 POINTS TO 'ITEM XX'.
5051 043026 012037 043036          MOV      (R0)+,$3          ; GET EMXX POINTER.
5052 043032 001404          BEQ      4$
5053 043034 104401          TYPE      EM TEXT.
5054 043036 000000          3$:    0
5055 043040 104401 001231          TYPE      , $CRLF
5056 043044 012037 043054          4$:    MOV      (R0)+,$5          ; GET DHXX POINTER.
5057 043050 001404          BEQ      6$
5058 043052 104401          TYPE      DH TEXT.
5059 043054 000000          5$:    0
5060 043056 104401 001231          TYPE      , $CRLF
5061
5062 043062 012001          6$:    MOV      (R0)+,R1          ; R1 POINTS TO 'DTXX'...
5063 043064 011000          MOV      (R0),R0          ; ...AND R0 TO 'DFXX'.
5064 043066 005711          ERT1:   TST      (R1)
5065 043070 001474          BEQ      ERT4          ; EXIT AT END OF DATA TABLE.
5066
5067 043072 105710          1$:    TSTB     (R0)          ; FORMAT 0 ??
5068 043074 001003          BNE      2$
5069 043076 013146          MOV      @ (R1)+,-(SP)          ; YES, TYPE AN OCTAL VALUE.
5070 043100 104402          TYPOC
5071 043102 000463          BR      ERT2
5072
5073 043104 121027 000002          2$:    CMPB     (R0),#2          ; FORMAT 2 ??
5074 043110 001010          BNE      3$
5075 043112 013102          MOV      @ (R1)+,R2          ; YES, TYPE 2 OCTAL VALUES.
5076 043114 012246          MOV      (R2)+,-(SP)
5077 043116 104402          TYPOC
5078 043120 104401 043300          TYPE      ,SPAC1
5079 043124 011246          MOV      (R2),-(SP)
5080 043126 104402          TYPOC
5081 043130 000450          BR      ERT2
5082
5083 043132 121027 000003          3$:    CMPB     (R0),#3          ; FORMAT 3 ??
5084 043136 001011          BNE      5$
    
```

5085	043140	012703	000004		MOV	#4,R3		; YES, TYPE 4 OCTAL VALUES.
5086	043144	013102			MOV	@(R1)+,R2		
5087	043146	012246		4\$:	MOV	(R2)+,-(SP)		
5088	043150	104402			TYPOC			
5089	043152	104401	043300		TYPE	,SPAC1		
5090	043156	077305			SOB	R3,1\$		
5091	043160	000434			BR	ERT2		
5092								
5093	043162	121027	000004	5\$:	CMPB	(R0),#4		; FORMAT 4 ??
5094	043166	001004			BNE	6\$		
5095	043170	013146			MOV	@(R1)+,-(SP)		; YES, TYPE 4 OCTAL DIGITS...
5096	043172	104403			TYPOS			;...SUPPRESS LEAD ZEROS.
5097	043174	004	000		.BYTE	4,0		
5098	043176	000425			BR	ERT2		
5099								
5100	043200	121027	000005	6\$:	CMPB	(R0),#5		; FORMAT 5 ??
5101	043204	001005			BNE	10\$		
5102	043206	012137	043214		MOV	(R1)+,7\$; YES, TYPE ASCII STRING.
5103	043212	104401			TYPE			
5104	043214	000000		7\$:	0			
5105	043216	000417			BR	ERT3		
5106								
5107	043220	121027	000012	10\$:	CMPB	(R0),#12		; FORMAT 12 ??
5108	043224	001011			BNE	12\$		
5109	043226	012703	000010		MOV	#8,R3		; YES, TYPE 8 OCTAL VALUES.
5110	043232	013102			MOV	@(R1)+,R2		
5111	043234	012246		11\$:	MOV	(R2)+,-(SP)		
5112	043236	104402			TYPOC			
5113	043240	104401	043300		TYPE	,SPAC1		
5114	043244	077305			SOB	R3,11\$		
5115	043246	000401			BR	ERT2		
5116	043250	000000		12\$:	HALT			; UNDEFINED FORMAT CODE.
5117								
5118	043252	104401	043302	ERT2:	TYPE	,HTAB		; PRINT A TAB (AFTER DATA ONLY).
5119	043256	005200		ERT3:	INC	R0		; POINT TO NEXT FORMAT BYTE.
5120	043260	000702			BR	ERT1		
5121								
5122	043262	104401	001231	ERT4:	TYPE	,\$CRLF		
5123	043266	012603			MOV	(SP)+,R3		; RESTORE REGISTERS.
5124	043270	012602			MOV	(SP)+,R2		
5125	043272	012601			MOV	(SP)+,R1		
5126	043274	012600			MOV	(SP)+,R0		
5127	043276	000207			RTS	PC		
5128								
5129	043300	040	000	SPAC1:	.BYTE	40,0		
5130	043302	011	000	HTAB:	.BYTE	11,0		

```
5132 .SBITL ASCII TEXT AND ERROR MESSAGES.
5133 :
5134 : FIRST SOME BITS AND PIECES COMMON TO MANY.
5135 :
5136 :.NLIST BEX
5137 043304 050200 053517 051105 POWERM: .ASCIZ <CRLF>'POWER FAIL -- RESTARTING.'<CRLF>
5138 043340 000 NULL: .BYTE 0
5139 043341 101 047502 052122 ABORT: .ASCIZ 'ABORTING TEST... '<CRLF>
5140 043363 200 054105 042520 EAC: .ASCII <CRLF>'EXPECTED AC'
5141 043377 060 020072 000 EACN: .ASCIZ '0: '
5142 043403 200 042522 042503 RAC: .ASCII <CRLF>'RECEIVED AC'
5143 043417 060 020072 000 RACN: .ASCIZ '0: '
5144 043423 200 047523 051125 BSRC: .ASCIZ <CRLF>'SOURCE DATA (BEFORE): '
5145 043453 200 047523 051125 ASRC: .ASCIZ <CRLF>'SOURCE DATA (AFTER): '
5146 043503 200 051050 024460 ARZ: .ASCIZ <CRLF>'(RO) AFTER EXECUTION: '
5147 043533 200 051123 020103 SRC: .ASCIZ <CRLF>'SRC DATA: '
5148 043547 200 054105 042520 EXP: .ASCIZ <CRLF>'EXPECTED: '
5149 043563 200 042522 042503 RCD: .ASCIZ <CRLF>'RECEIVED: '
5150 043577 200 054105 042520 EAC15: .ASCIZ <CRLF>'EXPECTED AC1 THRU AC5: '
5151 043627 200 042522 042503 RAC15: .ASCIZ <CRLF>'RECEIVED AC1 THRU AC5: '
5152 043657 200 041501 035060 ACZ: .ASCIZ <CRLF>'ACO: '
5153 043666 042600 050130 041505 ERES: .ASCIZ <CRLF>'EXPECTED RESULT: '
5154 043713 200 042522 042503 RRES: .ASCIZ <CRLF>'RECEIVED RESULT: '
5155 043740 042600 050130 041505 ESTAT: .ASCIZ <CRLF>'EXPECTED FPS AND FEC: '
5156 043770 051200 041505 044505 RSTAT: .ASCIZ <CRLF>'RECEIVED FPS AND FEC: '
5157 044020 020200 041501 047440 ACOPR: .ASCIZ <CRLF>' AC OPERAND: '
5158 044037 200 051123 020103 SOPR: .ASCIZ <CRLF>'SRC OPERAND: '
5159 044056 042200 052123 047440 DOPR: .ASCIZ <CRLF>'DST OPERAND: '
5160 044075 200 041501 020060 ACOB: .ASCIZ <CRLF>'ACO BEFORE: '
5161 044113 200 041501 020060 ACOA: .ASCIZ <CRLF>'ACO AFTER: '
5162 044131 200 054105 042120 EFRAC: .ASCIZ <CRLF>'EXPD FRACTION (ACO): '
5163 044160 051200 041505 020104 RFRAC: .ASCIZ <CRLF>'RECD FRACTION: '
5164 044207 200 054105 042120 EINT: .ASCIZ <CRLF>'EXPD INTEGER (AC1): '
5165 044236 051200 041505 020104 RINT: .ASCIZ <CRLF>'RECD INTEGER: '
5166 044265 200 041520 020054 REGS: .ASCIZ <CRLF>'PC, FPS, RO, AND ACO '
5167 044314 042502 047506 042522 BI: .ASCIZ 'BEFORE INTERRUPT:'<CRLF>
5168 044337 101 052106 051105 AI: .ASCIZ 'AFTER INTERRUPT:'<CRLF>
5169 :
5170 : ERROR MESSAGES:
5171 :
5172 044361 114 043104 051520 EM1: .ASCIZ 'LDFPS AND STFPS FAILED.'
5173 044411 114 043104 051520 EM2: .ASCIZ 'LDFPS AND STFPS ERROR SUMMARY.'
5174 044450 051520 020127 047111 EM3: .ASCIZ 'PSW INCORRECT AFTER CFCC.'
5175 044502 050106 020123 047111 EM4: .ASCIZ 'FPS INCORRECT AFTER CFCC.'
5176 044534 047125 054105 042520 EM5: .ASCIZ 'UNEXPECTED FPP TRAP TO 244.'
5177 044570 047125 054105 042520 EM6: .ASCIZ 'UNEXPECTED CPU TRAP TO 4.'
5178 044622 047125 054105 042520 EM7: .ASCIZ 'UNEXPECTED CPU TRAP TO 10.'
5179 :
5180 044655 106 051520 044440 EM12: .ASCII 'FPS INCORRECT AFTER SET'
5181 044704 020111 047111 052123 EM12X: .ASCIZ 'I INSTRUCTION.'
5182 044723 111 046114 043505 EM16: .ASCIZ 'ILLEGAL FP OPCODE DID NOT TRAP.'
5183 044763 106 051520 044440 EM17: .ASCIZ 'FPS INCORRECT AFTER ILLEGAL OPCODE TRAP.'
5184 045034 042506 020103 047111 EM20: .ASCIZ 'FEC INCORRECT AFTER ILLEGAL OPCODE TRAP.'
5185 :
5186 045105 106 051520 044440 EM22: .ASCIZ 'FPS INCORRECT AFTER ILLEGAL OPCODE (INT DISABLED).'
5187 045170 042506 020103 047111 EM23: .ASCIZ 'FEC INCORRECT AFTER ILLEGAL OPCODE (INT DISABLED).'
```

5188	045253	111	046114	043505	EM24:	.ASCIZ	'ILLEGAL OPCODE TRAPPED WITH INTERRUPT DISABLED.'
5189	045333	123	052517	041522	EM25:	.ASCIZ	'SOURCE DATA ALTERED AFTER LDD (RO),ACO.'
5190	045403	122	020060	047111	EM26:	.ASCII	'RO INCORRECT AFTER LDD (RO),ACO.'
5191	045443	200	041050	052125		.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 2 OR 4.'
5192							
5193	045506	030122	044440	041516	EM31:	.ASCII	'RO INCORRECT AFTER STD ACO,(RO).'
5194	045546	024200	052502	020124		.ASCIZ	<CRLF>'(BUT FDST) FAILED TO MODE 2 OR 4.'
5195							
5196	045611	123	042124	040440	EM34:	.ASCII	'STD ACO,(RO) EXECUTED WRONG DST MODE.'
5197	045656	024200	052502	020124		.ASCIZ	<CRLF>'(BUT GR7) FAILED TO IMMEDIATE MODE.'
5198	045723	114	042104	024040	EM35:	.ASCII	'LDD (RO),ACO EXECUTED WRONG SRC MODE.'
5199	045770	024200	052502	020124		.ASCIZ	<CRLF>'(BUT GR7) FAILED TO IMMEDIATE MODE.'
5200	046035	123	042124	040440	EM36:	.ASCII	'STD ACO,(RO) EXECUTED WRONG DATA MODE.'
5201	046103	200	041050	052125		.ASCIZ	<CRLF>'(BUT FD) FAILED TO SINGLE-PRECISION.'
5202	046151	114	042104	024040	EM37:	.ASCII	'LDD (RO),ACO EXECUTED WRONG DATA MODE.'
5203	046217	200	041050	052125		.ASCIZ	<CRLF>'(BUT FD) FAILED TO SINGLE-PRECISION.'
5204	046265	104	052101	020101	EM40:	.ASCIZ	'DATA INCORRECT AFTER LDD AND STD.'
5205	046327	106	051520	044440	EM41:	.ASCIZ	'FPS INCORRECT AFTER LOAD AND/OR STORE.'
5206	046376	042114	020104	051050	EM42:	.ASCII	'LDD (RO),ACO TRAPPED TO 4.'<CRLF>
5207	046431	050	052502	020124		.ASCIZ	'(BUT FSRC) FAILED TO MODE 3, 5, 6, OR 7.'
5208	046502	052123	020104	041501	EM43:	.ASCII	'STD ACO,(RO) TRAPPED TO 4.'<CRLF>
5209	046535	050	052502	020124		.ASCIZ	'(BUT FDST) FAILED TO MODE 3, 5, 6, OR 7.'
5210	046606	041501	052503	052515	EM44:	.ASCIZ	'ACCUMULATORS DATA TEST FAILED.'
5211							
5212	046645	101	041503	046525	EM46:	.ASCIZ	'ACCUMULATORS DUAL ADDRESSING TEST FAILED.'
5213	046717	114	057504	040440	EM47:	.ASCII	'LD AC1,ACO TRAPPED TO 4.'<CRLF>
5214	046751	050	052502	020124		.ASCIZ	'(BUT FSRC) FORK FAILED IN FSRC FLOW.'
5215	047016	042114	020137	041501	EM50:	.ASCII	'LD AC1,ACO DATA FORMAT INCORRECT.'<CRLF>
5216	047061	050	052502	020124		.ASCIZ	'(BUT FD) FORK FAILED IN FSRC FLOW.'
5217	047124	042114	020137	041501	EM51:	.ASCIZ	'LD AC1,ACO TRANSFERRED BAD DATA.'
5218	047166	042114	020104	051050	EM52:	.ASCIZ	'LDD (RO)+,ACO TRAPPED TO 4.'
5219	047222	030122	044440	041516	EM53:	.ASCIZ	'RO INCORRECT AFTER LDD (RO)+,ACO.'
5220	047264	040504	040524	044440	EM54:	.ASCIZ	'DATA INCORRECT AFTER LDD (RO)+,ACO.'
5221	047330	042114	020104	024055	EM55:	.ASCIZ	'LDD -(RO),ACO TRAPPED TO 4.'
5222	047364	030122	044440	041516	EM56:	.ASCIZ	'RO INCORRECT AFTER LDD -(RO),ACO.'
5223	047426	040504	040524	044440	EM57:	.ASCIZ	'DATA INCORRECT AFTER LDD -(RO),ACO.'
5224	047472	030122	044440	041516	EM60:	.ASCIZ	'RO INCORRECT AFTER LDF (RO)+,ACO.'
5225	047534	040504	040524	044440	EM61:	.ASCIZ	'DATA INCORRECT AFTER LDF (RO)+,ACO.'
5226	047600	040504	040524	043040	EM62:	.ASCII	'DATA FORMAT INCORRECT AFTER LDF (RO)+,ACO.'
5227	047652	024200	052502	020124		.ASCIZ	<CRLF>'(BUT FD) FORK FAILED IN FSRC FLOW.'
5228	047716	040504	040524	044440	EM63:	.ASCII	'DATA INCORRECT AFTER LDD #5204,ACO.'
5229	047761	200	041050	052125		.ASCIZ	<CRLF>'(BUT GR7) FORK FAILED IN FSRC FLOW.'
5230	050026	041520	044440	041516	EM64:	.ASCIZ	'PC INCORRECT AFTER LDD #5204,ACO.'
5231	050070	042114	020104	032443	EM65:	.ASCIZ	'LDD #5204,ACO TRAPPED TO 4.'
5232	050124	040504	040524	044440	EM66:	.ASCIZ	'DATA INCORRECT AFTER LDD #5204,ACO.'
5233	050170	042114	020104	024100	EM67:	.ASCII	'LDD @(RO)+,ACO TRAPPED TO 4.'
5234	050224	024200	052502	020124		.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 5, 6, OR 7.'
5235		050170			EM70=EM67		
5236	050273	114	042104	040040	EM71:	.ASCII	'LDD @(RO)+,ACO EXECUTED WRONG FSRC MODE.'
5237	050343	200	041050	052125		.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 0, 1, 2, OR 4.'
5238	050415	122	020060	047111	EM72:	.ASCIZ	'RO INCORRECT AFTER LDD @(RO)+,ACO.'
5239	050460	040504	040524	044440	EM73:	.ASCIZ	'DATA INCORRECT AFTER LDD @(RO)+,ACO.'
5240	050525	114	042104	040040	EM74:	.ASCII	'LDD @-(RO),ACO TRAPPED TO 4.'
5241	050561	200	041050	052125		.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 3, 6, OR 7.'
5242		050525			EM75-EM74		
5243	050630	042114	020104	026500	EM76:	.ASCII	'LDD @-(RO),ACO EXECUTED WRONG FSRC MODE.'

5244	050700	024200	052502	020124	.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 0, 1, 2, OR 4.'
5245	050752	030122	044440	041516	EM77:	.ASCIZ 'RO INCORRECT AFTER LDD @-(RO),ACO.'
5246	051015	104	052101	020101	EM100:	.ASCIZ 'DATA INCORRECT AFTER LDD @-(RO),ACO.'
5247	051062	042114	020104	032062	EM101:	.ASCII 'LDD 241(RO),ACO TRAPPED TO 4.'
5248	051117	200	041050	052125	.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO EXECUTE MODE 6.'
5249	051165	122	020060	047111	EM102:	.ASCIZ 'RO INCORRECT AFTER LDD 241(RO),ACO.'
5250	051231	114	042104	031040	EM103:	.ASCII 'LDD 241(RO),ACO EXECUTED WRONG FSRC MODE.'
5251	051302	024200	052502	020124	.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 0.'
5252	051340	040504	040524	044440	EM104:	.ASCIZ 'DATA INCORRECT AFTER LDD 241(RO),ACO.'
5253	051406	042114	020104	031100	EM105:	.ASCII 'LDD @241(RO),ACO TRAPPED TO 4.'
5254	051444	024200	052502	020124	.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO EXECUTE MODE 7.'
5255	051512	030122	044440	041516	EM106:	.ASCIZ 'RO INCORRECT AFTER LDD @241(RO),ACO.'
5256	051557	114	042104	040040	EM107:	.ASCII 'LDD @241(RO),ACO EXECUTED WRONG FSRC MODE.'
5257	051631	200	041050	052125	.ASCIZ	<CRLF>'(BUT FSRC) FAILED TO MODE 0 OR 6.'
5258	051674	040504	040524	044440	EM110:	.ASCIZ 'DATA INCORRECT AFTER LDD @241(RO),ACO.'
5259	051743	114	042104	040440	EM111:	.ASCII /LDD AC/
5260	051751	066	040454	030103	EM111X:	.ASCIZ /6,ACO DIDN'T TRAP (ILLEGAL FSRC AC)./
5261	052016	042101	042104	040440	EM112:	.ASCII /ADDD AC/
5262	052025	066	040454	030103	EM112X:	.ASCIZ /6,ACO DIDN'T TRAP (ILLEGAL FSRC AC)./
5263	052072	050106	020123	047111	EM113:	.ASCIZ 'FPS INCORRECT AFTER ILLEGAL AC TRAP.'
5264	052137	106	041505	044440	EM114:	.ASCIZ 'FEC INCORRECT AFTER ILLEGAL AC TRAP.'
5265	052204	046111	042514	040507	EM115:	.ASCIZ 'ILLEGAL AC REFERENCE TRAPPED TO 4 INSTEAD OF 244.'
5266						
5267	052266	052123	020137	041501	EM121:	.ASCII 'ST_ACO,AC1 TRAPPED TO 4.'<CRLF>
5268	052320	041050	052125	043040	.ASCIZ	'(BUT FDST) FORK FAILED IN FDST FLOW.'
5269	052365	123	057524	040440	EM122:	.ASCII 'ST_ACO,AC1 DATA FORMAT INCORRECT.'<CRLF>
5270	052430	041050	052125	043040	.ASCIZ	'(BUT FD) FORK FAILED IN FDST FLOW.'
5271	052473	123	057524	040440	EM123:	.ASCIZ 'ST_ACO,AC1 TRANSFERRED BAD DATA.'
5272						
5273	052535	106	051520	044440	EM125:	.ASCII 'FPS INCORRECT AFTER LDD (RO),ACO.'<CRLF>
5274	052577	050	052502	020124	.ASCIZ	'(BUT FZ) OR (BUT FN) FLOW FAILED.'
5275	052641	125	042116	043105	EM126:	.ASCII 'UNDEFINED VARIABLE (-0) TRAP DID NOT OCCUR.'
5276	052714	040600	030103	036440	.ASCIZ	<CRLF>'ACO = -0 AND FIUV = 1 (INT ENABLED).'
5277	052762	047125	042504	044506	EM127:	.ASCII 'UNDEFINED VARIABLE (-0) TRAPPED TO 244.'
5278	053031	200	041501	020060	.ASCIZ	<CRLF>'ACO = -0 AND FIUV = 0 (INT DISABLED).'
5279	053100	042506	020103	047111	EM130:	.ASCIZ 'FEC INCORRECT AFTER FIUV (-0) TRAP.'
5280						
5281	053144	042522	052523	052114	EM140:	.ASCII 'RESULT AND/OR STATUS INCORRECT ON ADD'
5282	053211	104	024040	030122	EM140X:	.ASCIZ 'D (RO),ACO.'
5283	053225	122	051505	046125	EM141:	.ASCII 'RESULT AND/OR STATUS INCORRECT ON SUB'
5284	053272	020104	051050	024460	EM141X:	.ASCIZ 'D (RO),ACO.'
5285						
5286	053306	052523	020115	047111	EM145:	.ASCIZ 'SUM INCORRECT ON ADD (RO),ACO.'
5287	053345	101	042104	020104	EM146:	.ASCIZ 'ADDD (RO),ACO FAILED TO TRUNCATE (FT 1).'
5288	053420	042101	057504	024040	EM147:	.ASCIZ 'ADD (RO),ACO FAILED TO ROUND (FT - 0).'
5289	053470	042101	042104	024040	EM150:	.ASCIZ 'ADDD (RO),ACO ROUNDED IN F MODE.'
5290	053531	101	042104	020106	EM151:	.ASCIZ 'ADDF (RO),ACO ROUNDED IN D MODE.'
5291	053572	052523	020115	047111	EM152:	.ASCII 'SUM INCORRECT ON ADD'
5292	053616	020104	051050	024460	EM152X:	.ASCII 'D (RO),ACO.'<CRLF>
5293	053632	051120	041117	041101	.ASCIZ	'PROBABLE FAULT IN (FDST) EXPONENT ALIGNMENT FLOW.'
5294	053714	052523	020115	047111	EM153:	.ASCII 'SUM INCORRECT ON ADD'
5295	053740	020104	051050	024460	EM153X:	.ASCII 'D (RO),ACO.'<CRLF>
5296	053754	051120	041117	041101	.ASCIZ	'PROBABLE FAULT IN (FSRC) EXPONENT ALIGNMENT FLOW.'
5297	054036	052523	020115	047111	EM154:	.ASCIZ 'SUM INCORRECT ON ADDD (RO),ACO.'
5298	054076	044504	043106	051105	EM155:	.ASCIZ 'DIFFERENCE INCORRECT ON SUBD (RO),ACO.'
5299	054145	122	051505	046125	EM156:	.ASCII 'RESULT INCORRECT AFTER '

```

5300 054174 042101 042104 024040 EM156X: .ASCII 'ADDD (RO),ACO.'<CRLF>
5301 054213 120 047522 040502 .ASCIZ 'PROBABLE FAULT IN NORMALIZE FLOW.'
5302
5303 : THE REST OF THESE WERE FORMALLY IN PART 2 (OF 3).
5304 :
5305 054255 123 046525 047440 EM160: .ASCII 'SUM OR STATUS INCORRECT ON ADDD (RO),ACO.'
5306 054326 050200 047522 040502 .ASCIZ <CRLF>'PROBABLE FAULT IN ROUND/TRUNCATE FLOW.'
5307 054376 052523 020115 051117 EM162: .ASCIZ 'SUM OR FPS INCORRECT ON FORCED OVERFLOW.'
5308 054447 106 051117 042503 EM163: .ASCIZ 'FORCED OVERFLOW ON ADDD TRAPPED (FIV = 0).'
5309 054522 047506 041522 042105 EM164: .ASCIZ 'FORCED OVERFLOW ON ADDD DID NOT TRAP (FIV = 1).'
5310 054602 050106 020123 051117 EM165: .ASCIZ 'FPS OR FEC INCORRECT ON ADDD OVERFLOW TRAP.'
5311 054656 052523 020115 051117 EM166: .ASCIZ 'SUM OR FPS INCORRECT ON FORCED UNDERFLOW.'
5312 054730 047506 041522 042105 EM167: .ASCIZ 'FORCED UNDERFLOW ON ADDD TRAPPED (FIU = 0).'
5313 055004 047506 041522 042105 EM170: .ASCIZ 'FORCED UNDERFLOW ON ADDD DID NOT TRAP (FIU = 1).'
5314 055065 106 051520 047440 EM171: .ASCIZ 'FPS OR FEC INCORRECT ON ADDD UNDERFLOW TRAP.'
5315 055142 030122 044440 041516 EM172: .ASCII 'RO INCORRECT AFTER LDC'<177>
5316 .=-1
5317 .EVEN
5318 055170 042106 024040 030122 EM172X: .ASCIZ 'FD (RO)+,ACO.'
5319 055206 040504 040524 044440 EM173: .ASCII 'DATA INCORRECT AFTER LDC'<177>
5320 .=-1
5321 .EVEN
5322 055236 042106 024040 030122 EM173X: .ASCIZ 'FD (RO)+,ACO.'
5323
5324 055254 050106 020123 047111 EM201: .ASCII 'FPS INCORRECT AFTER CMPD (RO),ACO '
5325 055316 044527 044124 024040 .ASCII 'WITH (FSRC)'<177>
5326 .=-1
5327 .EVEN
5328 055332 036440 024040 041501 EM201X: .ASCIZ ' - (AC).'
5329 055343 101 030103 053440 EM202: .ASCIZ 'ACO WAS ALTERED BY CMPD (RO),ACO.'
5330 055405 106 051520 047440 EM203: .ASCIZ 'FPS OR FEC INCORRECT AFTER DIVIDE BY 0 ERROR.'
5331 055463 104 053111 042111 EM204: .ASCIZ 'DIVIDE BY 0 ERROR FAILED TO TRAP (FID = 0).'
5332 055537 104 053111 042111 EM205: .ASCIZ 'DIVIDE BY 0 ERROR TRAPPED WITH FID = 1.'
5333 055607 122 051505 046125 EM206: .ASCIZ 'RESULT OR FPS INCORRECT AFTER DIVF (RO),ACO.'
5334 055664 042522 052523 052114 EM207: .ASCII 'RESULT OR FPS INCORRECT AFTER DIVD (RO),ACO.'
5335 055740 026400 020055 047506 EM207X: .ASCIZ '<0>'-- FORCED OVER\UNDERFLOW.'
5336 055773 122 051505 046125 EM210: .ASCIZ 'RESULT OR FPS INCORRECT AFTER MULF (RO),ACO.'
5337 056050 042522 052523 052114 EM211: .ASCII 'RESULT OR FPS INCORRECT AFTER MULD (RO),ACO.'
5338 056124 026400 020055 047506 EM211X: .ASCIZ '<0>'-- FORCED OVER\UNDERFLOW.'
5339
5340 056157 122 051505 046125 EM220: .ASCIZ 'RESULTS OR FPS INCORRECT ON MODF (RO),ACO.'
5341 056232 042522 052523 052114 EM221: .ASCII 'RESULTS OR FPS INCORRECT ON MODD (RO),ACO.'
5342 056304 026400 020055 047506 EM221X: .ASCIZ '<0>'-- FORCED OVER\UNDERFLOW.'
5343 .EVEN
5344
5345 : DATA HEADERS
5346 :
5347 056340 042524 052123 004456 DH1: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5348 056370 053411 047522 042524 .ASCIZ '<TAB>'WROTE.'<TAB>'READ.'<TAB>'EXPECTED.'
5349 056420 042524 052123 004456 DH2: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERR COUNT'
5350 056450 041011 042101 042040 .ASCIZ '<TAB>'BAD DATA (AND)'<TAB>'BAD DATA (OR)'
5351 056506 042524 052123 004456 DH3: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5352 056536 051011 040505 020104 .ASCIZ '<TAB>'READ PSW.'<TAB>'EXPECTED PSW.'
5353 056567 124 051505 027124 DH4: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5354 056617 011 051127 052117 .ASCIZ '<TAB>'WROTE FPS.'<TAB>'FPS AFTER CFCC.'
5355 056653 124 051505 027124 DH5: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'TRAP PC.'
  
```

```

5356 056702 020011 050106 004523 .ASCIZ <TAB>'FPS'<TAB><TAB>'FEC'
5357 056716 042524 052123 004456 DH6: .ASCIZ 'TEST.'<TAB>'CALL PC.'<TAB>'TRAP PC.'
5358 056716 DH7=DH6
5359
5360 056746 042524 052123 004456 DH12: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5361 056776 051011 040505 020104 .ASCIZ <TAB>'READ FPS.'<TAB>'EXPECTED FPS.'
5362
5363 057027 124 051505 027124 DH16: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5364 057057 011 050117 041440 .ASCIZ <TAB>'OP CODE.'<TAB>'FPS.'
5365 056746 DH17=DH12
5366 057077 124 051505 027124 DH20: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5367 057127 011 042522 042101 .ASCIZ <TAB>'READ FEC'<TAB>'EXPECTED FEC'
5368
5369 056746 DH22=DH12
5370 057077 DH23=DH20
5371 057027 DH24=DH16
5372 057156 042524 052123 004456 DH25: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5373 057206 020011 051050 024460 .ASCIZ <TAB>'(R0)'
5374 057156 DH26=DH25
5375
5376 057156 DH31=DH25
5377
5378 057156 DH34=DH25
5379 057156 DH35=DH25
5380 057156 DH36=DH25
5381 057156 DH37=DH25
5382 057156 DH40=DH25
5383 056746 DH41=DH12
5384 057156 DH42=DH25
5385 057156 DH43=DH25
5386 057215 124 051505 027124 DH44: .ASCIZ 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5387
5388 057215 DH46=DH44
5389 057215 DH47=DH44
5390 057215 DH50=DH44
5391 057215 DH51=DH44
5392 057156 DH52=DH25
5393 057156 DH53=DH25
5394 057156 DH54=DH25
5395 057156 DH55=DH25
5396 057156 DH56=DH25
5397 057156 DH57=DH25
5398 057156 DH60=DH25
5399 057156 DH61=DH25
5400 057156 DH62=DH25
5401 057215 DH63=DH44
5402 057246 042524 052123 004456 DH64: .ASCII 'TEST.'<TAB>'CALL PC.'<TAB>'ERROR PC.'
5403 057276 051011 040505 020104 .ASCIZ <TAB>'READ PC'<TAB>'EXPECTED PC'
5404 057215 DH65=DH44
5405 057215 DH66=DH44
5406 057156 DH67=DH25
5407 057156 DH70=DH25
5408 057156 DH71=DH25
5409 057156 DH72=DH25
5410 057156 DH73=DH25
5411 057156 DH74=DH25

```

5412	057156			DH75=DH25	
5413	057156			DH76=DH25	
5414	057156			DH77=DH25	
5415	057156			DH100=DH25	
5416	057156			DH101=DH25	
5417	057156			DH102=DH25	
5418	057156			DH103=DH25	
5419	057156			DH104=DH25	
5420	057156			DH105=DH25	
5421	057156			DH106=DH25	
5422	057156			DH107=DH25	
5423	057156			DH110=DH25	
5424	057215			DH111=DH44	
5425	057215			DH112=DH44	
5426	056746			DH113=DH12	
5427	057077			DH114=DH20	
5428	056716			DH115=DH6	
5429					
5430	057215			DH121=DH44	
5431	057215			DH122=DH44	
5432	057215			DH123=DH44	
5433					
5434	056746			DH125=DH12	
5435	057215			DH126=DH44	
5436	057215			DH127=DH44	
5437	057077			DH130=DH20	
5438					
5439	056746			DH140=DH12	
5440	056746			DH141=DH12	
5441					
5442	057324	042524	052123	004456	DH145: .ASCII 'TEST.' <tab>'CALL PC.'<tab>'ERROR PC.'</tab></tab>
5443	057354	020011	050106	000123	.ASCIIZ <TAB>'FPS'
5444	057324				DH146=DH145
5445	057324				DH147=DH145
5446	057324				DH150=DH145
5447	057324				DH151=DH145
5448	057324				DH152=DH145
5449	057324				DH153=DH145
5450	057324				DH154=DH145
5451	057324				DH155=DH145
5452	057324				DH156=DH145
5453					
5454	056746				DH160=DH12
5455	056746				DH162=DH12
5456	056746				DH163=DH12
5457	056746				DH164=DH12
5458	056716				DH165=DH6
5459	056746				DH166=DH12
5460	056746				DH167=DH12
5461	056746				DH170=DH12
5462	056716				DH171=DH6
5463	057156				DH172=DH25
5464	057324				DH173=DH145
5465					
5466	056746				DH201=DH12
5467	056746				DH202=DH12


```

5468      057215      DH203=DH44
5469 057362 042524 052123 004456 DH204: .ASCII 'TEST.'

```

5524	057734				DT55=DT52	
5525	057532				DT56=DT26	
5526	057554				DT57=DT34	
5527	057532				DT60=DT26	
5528	057554				DT61=DT34	
5529	057554				DT62=DT34	
5530	057712				DT63=DT50	
5531	057450				DT64=DT2	
5532	057472				DT65=DT6	
5533	057712				DT66=DT50	
5534	057734				DT67=DT52	
5535	057734				DT70=DT52	
5536	057554				DT71=DT34	
5537	057532				DT72=DT26	
5538	057554				DT73=DT34	
5539	057734				DT74=DT52	
5540	057734				DT75=DT52	
5541	057554				DT76=DT34	
5542	057532				DT77=DT26	
5543	057554				DT100=DT34	
5544	057734				DT101=DT52	
5545	057532				DT102=DT26	
5546	057554				DT103=DT34	
5547	057554				DT104=DT34	
5548	057734				DT105=DT52	
5549	057532				DT106=DT26	
5550	057554				DT107=DT34	
5551	057554				DT110=DT34	
5552	057472				DT111=DT6	
5553	057472				DT112=DT6	
5554	057450				DT113=DT2	
5555	057450				DT114=DT2	
5556	057472				DT115=DT6	
5557						
5558	057472				DT121=DT6	
5559	057712				DT122=DT50	
5560	057712				DT123=DT50	
5561						
5562	057752	001160	001162	043302	DT125: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4,ACZ,\$TMP5,0
5563		057472			DT126=DT6	
5564		057472			DT127=DT6	
5565		057752			DT130=DT125	
5566						
5567	060000	001160	001162	043302	DT140: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4
5568	060020	044037	001172	044020	.WORD	SOPR,\$TMP5,ACOPR,\$TMP6,ERES,\$TMP7,RRES,\$TMP10,0
5569		060000			DT141=DT140	
5570						
5571	060042	001160	001162	043302	DT145: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,SOPR,\$TMP4
5572	060062	044020	001172	043666	.WORD	ACOPR,\$TMP5,ERES,\$TMP6,RRES,\$TMP7,0
5573		060042			DT146=DT145	
5574		060042			DT147=DT145	
5575		060042			DT150=DT145	
5576		060042			DT151=DT145	
5577		060042			DT152=DT145	
5578		060042			DT153=DT145	
5579		060042			DT154=DT145	

5580		060042			DT155=DT145	
5581		060042			DT156=DT145	
5582						
5583		060000			DT160=DT140	
5584		060000			DT162=DT140	
5585		060000			DT163=DT140	
5586		060000			DT164=DT140	
5587	060100	001160	001162	043302	DT165: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ESTAT,\$TMP3,\$TMP4,RSTAT,\$TMP5,\$TMP6,0
5588		060000			DT166=DT140	
5589		060000			DT167=DT140	
5590		060000			DT170=DT140	
5591		060100			DT171=DT165	
5592		057532			DT172=DT26	
5593	060126	001160	001162	043302	DT173: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,SRC,\$TMP4
5594	060146	043547	001172	043563	.WORD	EXP,\$TMP5,RCD,\$TMP6,0
5595						
5596	060160	001160	001162	043302	DT201: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4
5597	060200	044037	001172	044020	.WORD	SOPR,\$TMP5,ACOPR,\$TMP6,0
5598	060212	001160	001162	043302	DT202: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4
5599	060232	044075	001172	044113	.WORD	ACOB,\$TMP5,ACOA,\$TMP6,0
5600	060244	001160	001162	043302	DT203: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ESTAT,\$TMP3,RSTAT,\$TMP4,0
5601	060266	001160	001162	043302	DT204: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP7,HTAB,\$TMP10,0
5602		060266			DT205=DT204	
5603	060310	001160	001162	043302	DT206: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4
5604	060330	044020	001172	044037	.WORD	ACOPR,\$TMP5,SOPR,\$TMP6,ERES,\$TMP7,RRES,\$TMP10,0
5605		060310			DT207=DT206	
5606		060310			DT210=DT206	
5607		060310			DT211=DT206	
5608						
5609	060352	001160	001162	043302	DT220: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,HTAB,\$TMP4
5610	060372	044020	001172	044037	.WORD	ACOPR,\$TMP5,SOPR,\$TMP6,EFAC,\$TMP7,RFRAC,\$TMP10
5611	060412	044207	001202	044236	.WORD	EINT,\$TMP11,RINT,\$TMP12,0
5612		060352			DT221=DT220	
5613					.EVEN	
5614					:	
5615					: AND FINALLY, THE FORMAT CODES.	
5616					:	
5617	060424	004	000	005	DF1: .BYTE	4,0,5,0,5,0,0,0
5618	060434	004	000	005	DF2: .BYTE	4,0,5,4,5,0,5,0
5619	060444	004	000	005	DF3: .BYTE	4,0,5,0,5,0,5,0
5620		060444			DF4=DF3	
5621		060444			DF5=DF3	
5622	060454	004	000	005	DF6: .BYTE	4,0,5,0
5623		060454			DF7=DF6	
5624						
5625		060444			DF12=DF3	
5626						
5627		060444			DF16=DF3	
5628		060444			DF17=DF3	
5629		060444			DF20 DF3	
5630						
5631		060444			DF22=DF3	
5632		060444			DF23=DF3	
5633		060444			DF24=DF3	
5634	060460	004	000	005	DF25: .BYTE	4,0,5,0,5,0,5,3,5,3
5635		060444			DF26=DF3	

5692	060472				DF123=DF44	
5693						
5694	060542	004	000	005	DF125: .BYTE	4,0,5,0,5,0,5,0,5,3
5695	060454				DF126=DF6	
5696	060454				DF127=DF6	
5697	060542				DF130=DF125	
5698						
5699	060554	004	000	005	DF140: .BYTE	4,0,5,0,5,0,5,0,5,3,5,3,5,3,5,3
5700	060554				DF141=DF140	
5701						
5702	060574	004	000	005	DF145: .BYTE	4,0,5,0,5,0,5,3,5,3,5,3,5,3
5703	060574				DF146=DF145	
5704	060574				DF147=DF145	
5705	060574				DF150=DF145	
5706	060574				DF151=DF145	
5707	060574				DF152=DF145	
5708	060574				DF153=DF145	
5709	060574				DF154=DF145	
5710	060574				DF155=DF145	
5711	060574				DF156=DF145	
5712						
5713	060554				DF160=DF140	
5714	060554				DF162=DF140	
5715	060554				DF163=DF140	
5716	060554				DF164=DF140	
5717	060612	004	000	005	DF165: .BYTE	4,0,5,0,5,0,0,5,0,0
5718	060554				DF166=DF140	
5719	060554				DF167=DF140	
5720	060554				DF170=DF140	
5721	060612				DF171=DF165	
5722	060444				DF172=DF3	
5723	060624	004	000	005	DF173: .BYTE	4,0,5,0,5,0,5,3,5,3,5,3
5724						
5725	060640	004	000	005	DF201: .BYTE	4,0,5,0,5,0,5,0,5,3,5,3
5726	060640				DF202=DF201	
5727	060654	004	000	005	DF203: .BYTE	4,0,5,0,5,2,5,2
5728	060444				DF204=DF3	
5729	060444				DF205=DF3	
5730	060664	004	000	005	DF206: .BYTE	4,0,5,0,5,0,5,0,5,2,5,2,5,2,5,2
5731	060704	004	000	005	DF207: .BYTE	4,0,5,0,5,0,5,0,5,3,5,3,5,3,5,3
5732	060664				DF210=DF206	
5733	060704				DF211=DF207	
5734						
5735	060724	004	000	005	DF220: .BYTE	4,0,5,0,5,0,5,0,5,2,5,2,5,2,5,2,5,2
5736	060750	004	000	005	DF221: .BYTE	4,0,5,0,5,0,5,0,5,3,5,3,5,3,5,3,5,3
5737					.EVEN	

AADONE	020056	AFATAL=	C00000	BB16	021526	CCP4	021004	DD7	022432
AAOP0	017766	AI	044337	BB17	021560	CCP5	021014	DD8	022466
AAGP1	017776	AMADR1=	000000	BB2	021144	CCP6	021024	DF1	060424
AAOP2	020006	AMADR2=	000000	BB20	021630	CCP7	021034	DF100 =	060460
AAOP3	020016	AMADR3=	000000	BB21	021666	CCSUM	020734	DF101 =	060534
AAOP4	020026	AMADR4=	000000	BB26	021736	CC1	020064	DF102 =	060444
AAOP5	020036	AMAMS1=	000000	BB27	022000	CC13	020276	DF103 =	060460
AAOP6	020046	AMAMS2=	000000	BB3	021212	CC14	020342	DF104 =	060460
AASUM	017756	AMAMS3=	000000	BB32	022046	CC19	020412	DF105 =	060534
AA1	017326	AMAMS4=	000000	BB33	022102	CC2	020126	DF106 =	060444
AA10	017460	AMSGAD=	000000	BB4	021242	CC20	020450	DF107 =	060460
AA11	017520	AMSGLG=	000000	BB6	021310	CC25	020520	DF110 =	060460
AA2	017400	AMSGTY=	000000	BB7	021344	CC26	020562	DF111 =	060454
AA20	017616	AMTYP1=	000000	BDONE	004652	CC31	020630	DF112 =	060454
AA21	017662	AMTYP2=	000000	BI	044314	CC32	020664	DF113 =	060444
ABASE =	000000	AMTYP3=	000000	BIT0	- 000001	CC7	020174	DF114 =	060444
ABORT	043341	AMTYP4=	000000	BIT00 =	000001	CC8	020230	DF115 =	060454
ACDW1 =	000000	ANALYZ	060774	BIT01 =	000002	CDONE	005130	DF12	= 060444
ACDW2 =	000000	APASS =	000000	BIT02 =	000004	CHECK2	004302	DF121 =	060454
ACOPR	044020	APRIOR=	000000	BIT03 =	000010	CHECK4	004312	DF122 =	060472
ACPUOP=	000000	APTC SU=	000040	BIT04 =	000020	CKSWR =	104407	DF123 =	060472
ACZ	043657	APTENV=	000001	BIT05 =	000040	CLRFPS=	104412	DF125	060542
ACO =	%000000	APTSIZ=	000200	BIT06 =	000100	CR =	000015	DF126 =	060454
ACOA	044113	APTSP0=	000100	BIT07 =	000200	CRLF =	000200	DF127 =	060454
ACOB	044075	ARZ	043503	BIT08 =	000400	C1	005022	DF130 =	060542
AC1 =	%000001	ASRC	043453	BIT09 =	001000	C15	005036	DF140	060554
AC2 =	%000002	ASWREG=	000000	BIT1	- 000002	C25	005076	DF141 =	060554
AC3 =	%000003	ATESTN=	000000	BIT10 =	002000	DDDONE	023334	DF145	060574
AC4 =	%000004	AUNIT =	C00000	BIT11 =	004000	DDISP =	177570	DF146 =	060574
AC5 =	%000005	AUSWR =	000000	BIT12 =	010000	DDONE	005316	DF147 =	060574
AC6 =	%000006	AVECT1	000000	BIT13 =	020000	DDP0	023214	DF150 =	060574
AC7 =	%000007	AVECT2=	000000	BIT14 =	040000	DDP1	023224	DF151 =	060574
ADDW0 =	000000	A1	004404	BIT15 =	100000	DDP2	023234	DF152 =	060574
ADDW1 =	000000	A2	004406	BIT2 =	000004	DDP3	023244	DF153 =	060574
ADDW10=	000000	A3	004520	BIT3 =	000010	DDP4	023254	DF154 =	060574
ADDW11=	000000	A4	004526	BIT4 =	000020	DDP5	023264	DF155 =	060574
ADDW12=	000000	BBDONE	022322	BIT5 =	000040	DDP6	023274	DF156 =	060574
ADDW13=	000000	BBP0	022162	BIT6 =	000100	DDP7	023304	DF16	= 060444
ADDW14=	000000	BBPOA	022172	BIT7 =	000200	DDP8	023314	DF160 =	060554
ADDW15	000000	BBP1	022202	BIT8 =	000400	DDP9	023324	DF162 =	060554
ADDW2 =	000000	BBP10	022272	BIT9 =	001000	DDSUM	023204	DF163 =	060554
ADDW3 =	000000	BBP11	022302	BPTVEC=	000014	DD1	022330	DF164 =	060554
ADDW4 =	000000	BBP12	022312	BSRC	043423	DD12	022526	DF165	060612
ADDW5 =	000000	BBP2	022212	B1	004556	DD13	022562	DF166 =	060554
ADDW6 =	000000	BBP2A	022222	B2	004600	DD16	022622	DF167 =	060554
ADDW7 =	000000	BBP3	022232	CCDONE	021074	DD17	022656	DF17	= 060444
ADDW8 =	000000	BBP4	022242	CCP0	020744	DD2	022372	DF170 =	060554
ADDW9 =	000000	BBP5	022252	CCP1	020754	DD22	022716	DF171 =	060612
ADEVCT=	000000	BBP7	022262	CCP10	021044	DD23	022752	DF172 =	060444
ADEVN =	000000	BBSUM	022152	CCP11	021054	DD30	023012	DF173	060624
ADONE	004534	BB1	021102	CCP12	021064	DD31	023046	DF2	060434
AENV	000000	BB14	021412	CCP2	020764	DD36	023106	DF20 =	060444
AENVN =	000000	BB15	021456	CCP3	020774	DD37	023142	DF201	060640

DF202 = 060640	DF75 = 060534	DH201 = 056746	DH74 = 057156	DT171 = 060100
DF203 = 060654	DF76 = 060460	DH202 = 056746	DH75 = 057156	DT172 = 057552
DF204 = 060444	DF77 = 060444	DH203 = 057215	DH76 = 057156	DT173 = 060126
DF205 = 060444	DH1 = 056340	DH204 = 057362	DH77 = 057156	DT2 = 057450
DF206 = 060664	DH100 = 057156	DH205 = 057362	DISPLA = 001142	DT20 = 057450
DF207 = 060704	DH101 = 057156	DH206 = 056746	DISPRE = 000174	DT201 = 060160
DF210 = 060664	DH102 = 057156	DH207 = 056746	CJPR = 044056	DT202 = 060212
DF211 = 060704	DH103 = 057156	DH210 = 056746	DSWR = 177570	DT203 = 060244
DF22 = 060444	DH104 = 057156	DH211 = 056746	DT1 = 057426	DT204 = 060266
DF220 = 060724	DH105 = 057156	DH22 = 056746	DT100 = 057554	DT205 = 060266
DF221 = 060750	DH106 = 057156	DH220 = 056746	DT101 = 057734	DT206 = 060310
DF23 = 060444	DH107 = 057156	DH221 = 056746	DT102 = 057532	DT207 = 060310
DF24 = 060444	DH110 = 057156	DH23 = 057077	DT103 = 057554	DT210 = 060310
DF25 = 060460	DH111 = 057215	DH24 = 057027	DT104 = 057554	DT211 = 060310
DF26 = 060444	DH112 = 057215	DH25 = 057156	DT105 = 057734	DT22 = 057450
DF3 = 060444	DH113 = 056746	DH26 = 057156	DT106 = 057532	DT220 = 060352
DF31 = 060444	DH114 = 057077	DH3 = 056506	DT107 = 057554	DT221 = 060352
DF34 = 060460	DH115 = 056716	DH31 = 057156	DT110 = 057554	DT23 = 057450
DF35 = 060460	DH12 = 056746	DH34 = 057156	DT111 = 057472	DT24 = 057450
DF36 = 060460	DH121 = 057215	DH35 = 057156	DT112 = 057472	DT25 = 057504
DF37 = 060460	DH122 = 057215	DH36 = 057156	DT113 = 057450	DT26 = 057532
DF4 = 060444	DH123 = 057215	DH37 = 057156	DT114 = 057450	DT3 = 057450
DF40 = 060460	DH125 = 056746	DH4 = 056567	DT115 = 057472	DT31 = 057532
DF41 = 060444	DH126 = 057215	DH40 = 057156	DT12 = 057450	DT34 = 057554
DF42 = 060444	DH127 = 057215	DH41 = 056746	DT121 = 057472	DT35 = 057554
DF43 = 060444	DH130 = 057077	DH42 = 057156	DT122 = 057712	DT36 = 057554
DF44 = 060472	DH140 = 056746	DH43 = 057156	DT123 = 057712	DT37 = 057554
DF46 = 060502	DH141 = 056746	DH44 = 057215	DT125 = 057752	DT4 = 057450
DF47 = 060454	DH145 = 057324	DH46 = 057215	DT126 = 057472	DT40 = 057554
DF5 = 060444	DH146 = 057324	DH47 = 057215	DT127 = 057472	DT41 = 057450
DF50 = 060472	DH147 = 057324	DH5 = 056653	DT130 = 057752	DT42 = 057532
DF51 = 060472	DH150 = 057324	DH50 = 057215	DT140 = 060000	DT43 = 057532
DF52 = 060534	DH151 = 057324	DH51 = 057215	DT141 = 060000	DT44 = 057602
DF53 = 060444	DH152 = 057324	DH52 = 057156	DT145 = 060042	DT46 = 057624
DF54 = 060460	DH153 = 057324	DH53 = 057156	DT146 = 060042	DT47 = 057472
DF55 = 060534	DH154 = 057324	DH54 = 057156	DT147 = 060042	DT5 = 057450
DF56 = 060444	DH155 = 057324	DH55 = 057156	DT150 = 060042	DT50 = 057712
DF57 = 060460	DH156 = 057324	DH56 = 057156	DT151 = 060042	DT51 = 057712
DF6 = 060454	DH16 = 057027	DH57 = 057156	DT152 = 060042	DT52 = 057734
DF60 = 060444	DH160 = 056746	DH6 = 056716	DT153 = 060042	DT53 = 057532
DF61 = 060460	DH162 = 056746	DH60 = 057156	DT154 = 060042	DT54 = 057554
DF62 = 060460	DH163 = 056746	DH61 = 057156	DT155 = 060042	DT55 = 057734
DF63 = 060472	DH164 = 056746	DH62 = 057156	DT156 = 060042	DT56 = 057532
DF64 = 060444	DH165 = 056716	DH63 = 057215	DT16 = 057450	DT57 = 057554
DF65 = 060454	DH166 = 056746	DH64 = 057246	DT160 = 060000	DT6 = 057472
DF66 = 060472	DH167 = 056746	DH65 = 057215	DT162 = 060000	DT60 = 057532
DF67 = 060534	DH17 = 056746	DH66 = 057215	DT163 = 060000	DT61 = 057554
DF7 = 060454	DH170 = 056746	DH67 = 057156	DT164 = 060000	DT62 = 057554
DF70 = 060534	DH171 = 056716	DH7 = 056716	DT165 = 060100	DT63 = 057712
DF71 = 060460	DH172 = 057156	DH70 = 057156	DT166 = 060000	DT64 = 057450
DF72 = 060444	DH173 = 057324	DH71 = 057156	DT167 = 060000	DT65 = 057472
DF73 = 060460	DH2 = 056420	DH72 = 057156	DT17 = 057450	DT66 = 057712
DF74 = 060534	DH20 = 057077	DH73 = 057156	DT170 = 060000	DT67 = 057734

DT7	=	057472	EM12X	044704	EM211X	056124	ERT1	043066	GGP1	026560
DT70	=	057734	EM121	052266	EM22	045105	ERT2	043252	GGP2	026570
DT71	=	057554	EM122	052365	EM220	056157	ERT3	043256	GGP3	026600
DT72	=	057532	EM123	052473	EM221	056232	ERT4	043262	GGP4	026610
DT73	=	057554	EM125	052535	EM221X	056304	ESTAT	043740	GGP6	026620
DT74	=	057734	EM126	052641	EM23	045170	EXP	043547	GGP7	026630
DT75	=	057734	EM127	052762	EM24	045253	E1	005334	GGSUM	026540
DT76	=	057554	EM130	053100	EM25	045333	E2	005354	GG1	025624
DT77	=	057532	EM140	053144	EM26	045403	E244	005432	GG2	026004
D1		005144	EM140X	053211	EM3	044450	FDATIO	006420	GG244	026526
D2		005174	EM141	053225	EM31	045506	FDATI1	006422	GG3	026172
D244		005242	EM141X	053272	EM34	045611	FDATI2	006424	GG4	026336
D3		005216	EM145	053306	EM35	045723	FDATI3	006426	GG5	026524
D4		005270	EM146	053345	EM36	046035	FDAT00	006440	GPAT	010546
EAC		043363	EM147	053420	EM37	046151	FDAT01	006442	GTSWR =	104406
EACN		043377	EM150	053470	EM4	044502	FDAT02	006444	G1	010052
EAC15		043577	EM151	053531	EM40	046265	FDAT03	006446	G2	010140
EDONE		005456	EM152	053572	EM41	046327	FDONE	006470	G3	010164
EEDIFF		023732	EM152X	053616	EM42	046376	FERR10	006220	G4	010204
EEDONE		024032	EM153	053714	EM43	046502	FERR20	006224	G5	010212
EEP1		023742	EM153X	053740	EM44	046606	FERR25	006316	HA1R	011272
EEP2		023752	EM154	054036	EM46	046645	FERR6	006100	HA1W	011222
EEP3		023762	EM155	054076	EM47	046717	FERR7	006150	HA2R	011302
EEP4		023772	EM156	054145	EM5	044534	FFDONE	024570	HA2W	011232
EEP5		024002	EM156X	054174	EM50	047016	FFP0	024500	HA3R	011312
EEP6		024012	EM16	044723	EM51	047124	FFP1	024510	HA3W	011242
EEP7		024022	EM160	054255	EM52	047166	FFP2	024520	HA4R	011322
EE1		023542	EM162	054376	EM53	047222	FFP3	024530	HA4W	011252
EE2		023444	EM163	054447	EM54	047264	FFP4	024540	HA5R	011332
EE3		023540	EM164	054522	EM55	047330	FFP5	024550	HA5W	011262
EE4		023634	EM165	054602	EM56	047364	FFP6	024560	HCLR	011204
EE5		023730	EM166	054656	EM57	047426	FFSUM	024470	HDAT1	011342
EFRAC		044131	EM167	054730	EM6	044570	FF1	024040	HDAT2	011352
EINT		044207	EM17	044763	EM60	047472	FF2	024162	HDAT3	011362
EMTVEC=		000030	EM170	055004	EM61	047534	FF3	024256 G	HDAT4	011372
EM1		044361	EM171	055065	EM62	047600	FF4	024372	HDAT5	011402
EM100		051015	EM172	055142	EM63	047716	FF5	024466	HDONE	011412
EM101		051062	EM172X	055170	EM64	050026	FPVEC =	000244	HHDONE	025616
EM102		051165	EM173	055206	EM65	050070	FXDAT0	006460	HHPO	025456
EM103		051231	EM173X	055236	EM66	050124	FXDAT1	006462	HHP1	025466
EM104		051340	EM2	044411	EM67	050170	FXDAT2	006464	HHP10	025576
EM105		051406	EM20	045034	EM7	044622	FXDAT3	006466	HHP11	025606
EM106		051512	EM201	055254	EM70 =	050170	F1	005464	HHP2	025476
EM107		051557	EM201X	055332	EM71	050273	F10	005746	HHP3	025506
EM110		051674	EM202	055343	EM72	050415	F11	005750	HHP4	025516
EM111		051743	EM203	055405	EM73	050460	F3	005536	HHP5	025526
EM111X		051751	EM204	055463	EM74	050525	F4	005540	HHP6	025536
EM112		052016	EM205	055537	EM75 =	050525	F6	005664	HHP7	025546
EM112X		052025	EM206	055607	EM76	050630	GDAT	010556	HHP8	025556
EM113		052072	EM207	055664	EM77	050752	GDONE	010566	HHP9	025566
EM114		052137	EM207X	055740	ERES	043666	GGDONE	026640	HHSUM	025446
EM115		052204	EM210	055773	ERRVEC=	000004	GGFEC	026536	HH1	024576
EM12		044655	EM211	056050	ERTYPE	042742	GGP0	026550	HH11	025046

MM12	025076	JJP1	027526	LASTAD=	061372	NERR4	013500	PP13	033310
MM17	025174	JJP2	027536	LASTEM=	000223	NNDONE	032722	PP14	033334
MM18	025224	JJP3	027546	LASTST=	000051	NNSUB	032542	PP2	032754
MM2	024626	JJP4	027556	LDATIO	012670	NNSUB1	032532	PP3	033000
MM21	025316	JJP5	027566	LDAT00	012700	NN1	031770	PP4	033024
MM22	025346	JJP6	027576	LDONE	012710	NN2	032030	PP5	033050
MM7	024720	JJP7	027606	LF =	000012	NN3	032070	PP6	033074
MM8	024750	JJP8	027616	LLDONE	031164	NN4	032130	PP7	033120
HSTOR	011060	JJ1	026646	LLP1	031144	NN5	032170	PP8	033144
HT =	000011	JJ17	027172	LLP2	031154	NN6	032230	PP9	033170
HTAB	043302	JJ18	027230	LL0	030514	NN7	032270	PRGSIZ=	000142
H1	010600	JJ2	026716	LL2	030646	NN8	032330	PR0 =	000000
H3	010646	JJ21	027304	LL244	031116	NN9	032430	PR1 -	000040
IDATIO	007262	JJ22	027334	LL4	030764	NPAT10	013560	PR2 =	000100
IDATI1	007264	JJ27	027410	LOOP	004146	NPAT20	013546	PR3 =	000140
IDATI2	007266	JJ28	027440	LPAT10	012650	NULL	043340	PR4 =	000200
IDATI3	007270	JJ29	027514	LPAT20	012660	N1	013160	PR5 =	000240
IDAT00	007252	JJ8	027016	LUPERR=	104411	N2	013232	PR6 =	000300
IDATO1	007254	JJ9	027070	L1	012456	N3	013234	PR7 =	000340
IDATO2	007256	J1	011764	L2	012530	N4	013236	PS =	177776
IDATO3	007260	J10	012126	L3	012532	N5	013276	PSW -	177776
IDONE	007272	J2	012022	L6	012646	ODATIO	014216	PURVEC=	000024
IERR0	007046	J3	012024	MDAT00	013142	ODAT00	014154	P1	014234
IERR1	007136	J4	012026	MDONE	013152	ODONE	014226	P2	014306
IERR2	007164	J7	012104	MERR3	013062	OERR0	014002	P3 =	014310
IERR3	007170	K =	000223	MMDONE	031762	OERR1	014076	P4	014312
IERR4	007206	KBUF0	012420	MMSUB	031600	OERR10	014030	QDATIO	015206
IOTVEC=	000020	KDATIO	012410	MM1	031172	OERR11	014042	QDAT00	015176
IPAT10	007232	KDAT00	012430	MM10	031456	OERR2	014106	QDONE	015216
IPAT11	007234	KDONE	012450	MM11	031502	OERR3	014116	QERRO	015022
IPAT12	007236	KERR0	012344	MM12	031526	OERR4	014126	QPAT10	015156
IPAT13	007240	KKDONE	030506	MM13	031552	OPAT10	014206	QPAT20	015166
IPAT20	007242	KKSUB	030326	MM2	031216	OPAT20	014172	QQDONE	034430
IPAT21	007244	KK1	027634	MM3	031242	OPAT21	014174	QQSUB	034250
IPAT22	007246	KK10	030164	MM4	031266	O1	013606	QQSUB1	034240
IPAT23	007250	KK11	030214	MM5	031312	O2	013660	QQ1	033576
I1	006476	KK12	030244	MM6	031336	O3	013662	QQ2	033636
I11	006666	KK13	030274	MM7	031362	O4	013664	QQ3	033676
I14	006746	KK2	027664	MM8	031406	O5	013724	QQ4	033736
I15	006750	KK3	027714	MM9	031432	PDATIO	014574	QQ5	033776
I16	006752	KK4	027744	MPAT10	013122	PDAT00	014604	QQ6	034036
I17	006764	KK5	027774	MPAT20	013132	PDONE	014614	QQ7	034136
I3	006574	KK6	030024	M1	012716	PERRO	014402	Q1	014622
I4	006576	KK7	030054	M2	012750	PIRQ -	177772	Q2	014674
I5	006600	KK8	030104	NDATIO	013570	PIRQVE=	000240	Q3 -	014676
JBUF0	012154	KK9	030134	NDAT00	013526	POWERM	043304	J4	014700
JDATIO	012164	KPAT0	012440	NDONE	013600	PPAT10	014564	RAC	043403
JDAT00	012174	K1	012222	NERR^	013354	PPDONE	033570	RACN	043417
JDATO	012204	K10	012364	NERR1	013450	PPSUB	033406	RAC15	043627
JDONE	012214	K2	012260	NERR10	013402	PP1	032730	RCD	043563
JERRO	012106	K3	012262	NERR11	013414	PP10	033214	RDCHR =	104410
JJDAT	027516	K4	012264	NERR2	013460	PP11	033240	REGS	044265
JJDONE	027626	K7	012342	NERR3	013470	PP12	033264	RESTAR	003542

RESVEC=	000010	S4	011616	TST32	020062	UERR3	015506	YOPO	017022
RFRAC	044160	TAB =	000011	TST33	021100	UERR4	015512	YOP1	017032
RINT	044236	TBITVE=	000014	TST34	022326	UPAT0	015550	YOP2	017042
RRES	043713	TDAT10	010034	TST35	023340	UPAT1	015560	YRES	017052
RSTAT	043770	TDAT11	010036	TST36	024036	UPAT2	015570	Y1	016630
R6	=%000006	TDAT12	010040	TST37	024574	UPAT3	015600	Y2	016656
R7	=%000007	TDAT13	010042	TST4	005134	UPAT4	015610	Y3	016700
SCOPEX	040144	TDATOC	010024	TST40	025622	UUDONE	037270	Y4	016736
SDAT	011632	TDATO1	010026	TST41	026644	UUSUB	037026	ZDONE	017320
SDONE	011756	TDATO2	010030	TST42	027632	UUSUB1	037016	ZOPO	017260
SKP1	= 000401	TDATO3	010032	TST43	030512	UU1	035424	ZOP1	017270
SKP2	= 000402	TDONE	010044	TST44	031170	UU10	036174	ZOP2	017300
SKP3	= 000403	TERR0	007620	TST45	031766	UU11	036244	ZRES	017310
SOPR	044037	TERR1	007710	TST46	032726	UU12	036314	Z1	017074
SPAC1	043300	TERR2	007736	TST47	033574	UU13	036364	Z2	017114
SRC	043533	TERR3	007760	TST5	005322	UU14	036434	Z3	017130
STACK	= 001100	TERR4	007742	TST50	034434	UU15	036504	Z4	017174
START	003542	TKVEC =	000060	TST51	035422	UU16	036624	\$APTHD	001000
STKMT=	177774	TPAT10	010004	TST6	005462	UU17	036674	\$ATYC	041340
STRP1	011642	TPAT11	010006	TST7	006474	UU2	035474	\$ATY1	041314
STRP2	011736	TPAT12	010010	TTDONE	035416	UU3	035544	\$ATY3	041322
SVPC	= 041314	TPAT13	010012	TTSUB	035160	UU4	035614	\$ATY4	041332
SWR	001140	TPAT20	010014	TT1	034436	UU5	035664	\$AUTOB	001134
SWREG	000176	TPAT21	010016	TT10	034766	UU6	035734	\$BASE	001310
SW0	= 000001	TPAT22	010020	TT11	035016	UU7	036004	\$BDADR	001122
SW00	= 000001	TPAT23	010022	TT12	035046	UU8	036054	\$BDDAT	001126
SW01	= 000002	TPVEC =	000064	TT13	035076	UU9	036124	\$BELL	001224
SW02	= 000004	TRAPVE=	000034	TT14	035126	U1	015236	\$CHARC	040566
SW03	= 000010	TRP04	004224	TT2	034466	U2	015246	\$CKSWR	041562
SW04	= 000020	TRP10	004234	TT3	034516	U244	015532	\$CMTAG	001100
SW05	= 000040	TRP244	004200	TT4	034546	U3	015416	\$CM3 =	000000
SW06	= 000100	TRTVEC=	000014	TT5	034576	U4	015424	\$CM4 =	000020
SW07	= 000200	TST1	004356	TT6	034626	U5	015440	\$CNTLG	042171
SW08	= 000400	TST10	007276	TT7	034656	WDONE	016206	\$CNTLU	042164
SW09	= 001000	TST11	010050	TT8	034706	WERR1	016026	\$CPUOP	001262
SW1	= 000002	TST12	010572	TT9	034736	WERR2	016050	\$CRLF	001231
SW10	= 002000	TST13	011416	TYPDS =	104405	WERR3	016026	\$DBLK	041304
SW11	= 004000	TST14	011762	TYPE =	104401	WERR4	016050	\$DEVCT	001244
SW12	= 010000	TST15	012220	TYPGC =	104402	WOP1	016156	\$DOAGN	037474
SW13	= 020000	TST16	012454	TYPON =	104404	WOP2	016166	\$DTBL	041274
SW14	= 040000	TST17	012714	TYPOS =	104403	WRES	016176	\$ENDAD	037464
SW15	= 100000	TST2	004540	T1	007300	W1	015626	\$ENDCT	037330
SW2	= 000004	TST20	013156	T11	007444	XDONE	016616	\$ENULL	037500
SW3	= 000010	TST21	013604	T14	007520	XERR1	016450	\$ENV	001254
SW4	= 000020	TST22	014232	T15	007522	XERR2	016456	\$ENVM	001255
SW5	= 000040	TST23	014620	T16	007524	XERR3	016450	\$EOP	037274
SW6	= 000100	TST24	015222	T23	007616	XERR4	016456	\$EOPCT	037322
SW7	= 000200	TST25	015624	T3	007376	YOP1	016556	\$ERFLG	001103
SW8	= 000400	TST26	016212	T4	007400	XOP2	016566	\$ERMAX	001115
SW9	= 001000	TST27	016622	T5	007402	XOP3	016576	\$ERROR	042542
S1	011462	TST3	004662	UDONE	015620	XRES	016606	\$ERRPC	001116
S2	011520	TST30	017066	UERR1	015462	X1	016214	\$ERRTB	001312
S3	011560	TST31	017324	UERR2	015502	YDONE	017062	\$ERTTL	001112

\$ESCAP 001222	\$MAIL 001234	\$PWJRDN 042370	\$TMP11 001202	\$TYPEC 040450
\$ETABL 001254	\$MAMS1 001264	\$PWRMG 042524	\$TMP12 001204	\$TYPEX 040570
\$ETEND 001312	\$MAMS2 001270	\$PWRUP 042442	\$TMP13 001206	\$TYPOC 040666
\$FATAL 001236	\$MAMS3 001274	\$QUES 001230	\$TMP14 001210	\$TYPON 040702
\$FFLG 041560	\$MAMS4 001300	\$RDCHR 042044	\$TMP15 001212	\$TYPOS 040642
\$FILLC 001156	\$MBADR 001002	\$RDSZ - 000001	\$TMP16 001214	\$UNIT 001246
\$FILLS 001155	\$MFLG 041556	\$RTNAD 037476	\$TMP17 001216	\$UNITM 001010
\$GDADR 001120	\$MNEW 042207	\$SAVR6 042540	\$TMP2 001164	\$USWR 001260
\$GDDAT 001124	\$MSGAD 001250	\$SCOPE 037504	\$TMP3 001166	\$VECT1 001304
\$GET42 037454	\$MSGLG 001252	\$SETUP= 000137	\$TMP4 001170	\$VECT2 001306
\$GTSWR 041632	\$MSGTY 001234	\$STUP = 177777	\$TMP5 001172	\$XOFF = 000023
\$HD = 000003	\$MSWR 042176	\$SVLAD 037750	\$TMP6 001174	\$XON = 000021
\$HIBTS 001000	\$MTYP1 001265	\$SVPC = 001000	\$TMP7 001176	\$XTSTR 037516
\$ICNT 001104	\$MTYP2 001271	\$SWR = 167400	\$TN = 000052	\$GET4= 000000
\$ILLUP 042534	\$MTYP3 001275	\$SWREG 001256	\$TPB 001152	\$SW08= 000052
\$INTAG 001135	\$MTYP4 001301	\$SWRMK= 000200	\$TPFLG 001157	\$TCX 040572
\$ITEMB 001114	\$MXCNT 040020	\$SWOBT 040022	\$TPS 001150	\$OFILL 041065
\$LF 001232	\$NULL 001154	\$TERM = 000026	\$TRAP 042220	. = 061372
\$LFLG 041557	\$NWTST= 000001	\$TESTN 001240	\$TRAP2 042242	.CLRFP 042310
\$LPADR 001106	\$OCNT 041064	\$TIMES 001220	\$TRP = 000013	.LPER 042302
\$LPERR 001110	\$OMODE 041066	\$TKB 001146	\$TRPAD 042254	.\$X = 001000
\$MADR1 001266	\$OVER 040004	\$TKS 001144	\$STM 001004	
\$MADR2 001272	\$PASS 001242	\$TMP0 001160	\$STNM 001102	
\$MADR3 001276	\$PASTM 001006	\$TMP1 001162	\$TYPDS 041070	
\$MADR4 001302	\$PWAD 042530	\$TMP10 001200	\$TYPE 040236	

. ABS. 061372 000 CON RO REL GBL D

ERRORS DETECTED: 0

CJFPAA,CJFPAA/LI:TOC=SYSMAC/ML,CJFPAA
 RUN-TIME: 36 21 1 SECONDS
 RUN-TIME RATIO: 555/59=9.3
 CORE USED: 27K (53 PAGES)