

M8188,
FPF11

FPF11 FLT PNT DIAG #2
CJFPBAO

AH-S444A-MC
FICHE 1 OF 1

OCT 1981
COPYRIGHT © 1981
MADE IN USA



The main body of the document is a large grid of small, illegible tables or diagrams. Each cell in the grid appears to contain technical data, possibly related to a flight point diagnosis, but the text is too small to be read. The grid is organized into approximately 10 columns and 20 rows.

IDENTIFICATION

PRODUCT CODE: AC-S442A-MC
PRODUCT NAME: CJFPBA0 FPF11 FLT PNT DIAG #2
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

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 2).
=====
- 9.1 TEST 1 -- STF (FDST MODE 0), USING AN ILLEGAL AC.
=====
TEST FDST MODE 0, USING STF AND AN ILLEGAL AC.
EXPECT AN ERROR TRAP.
- 9.2 TEST 2 -- STF (FDST MODE 1) INDIRECT.
=====
TEST FDST MODE 1 (INDIRECT) USING STF INSTRUCTION.
- 9.3 TEST 3 -- STF AND STD (FDST MODE 2) AUTO-INCREMENT.
=====
TEST FDST MODE 2 (AUTO-INCREMENT) USING STF AND STD.
- 9.4 TEST 4 -- STD (FDST MODE 2 WITH R7) PC IMMEDIATE.
=====
TEST FDST MODE 2, GR7 (PC IMMEDIATE) USING STD INSTRUCTION.
- 9.5 TEST 5 -- STD (FDST MODE 4) AUTO-DECREMENT.
=====
TEST FDST MODE 4 (AUTO-DECREMENT) USING STD INSTRUCTION.
- 9.6 TEST 6 -- STD (FDST MODE 3) AUTO-INCREMENT DEFERRED.
=====
TEST FDST MODE 3 (AUTO-INCREMENT DEFERRED) USING STD.
- 9.7 TEST 7 -- STD (FDST MODE 5) AUTO-DECREMENT DEFERRED.
=====
TEST FDST MODE 5 (AUTO-DECREMENT DEFERRED) USING STD.
- 9.10 TEST 10 -- STD (FDST MODE 6) INDEXED.
=====
TEST FDST MODE 6 (INDEX) USING STD.
- 9.11 TEST 11 -- STD (FDST MODE 7) INDEX DEFERRED.
=====
TEST FDST MODE 7 (INDEX DEFERRED) USING STD.

- 9.12 TEST 12 -- STCFD AND STCDF (FDST MODE 1).
 =====
 TESTS THE STORE CONVERTED (STCFD AND STCDF) INSTRUCTIONS.
- 9.13 TEST 13 -- STCFD (FDST MODE 0) USING AN ILLEGAL AC.
 =====
 TEST STCFD WITH AN ILLEGAL ACCUMULATOR (AC6).
 EXPECT AN ERROR TRAP.
- 9.14 TEST 14 -- CLRF AND CLRD (FDST MODE 1).
 =====
 TEST THE CLRF AND CLRD INSTRUCTIONS.
- 9.15 TEST 15 -- CLRD (FDST MODE 0) USING AN ILLEGAL AC.
 =====
 TEST CLRD WITH AN ILLEGAL ACCUMULATOR (AC7).
 EXPECT AN ERROR TRAP.
- 9.16 TEST 16 -- SPECIAL FDST MODE 0, USING NEGD AND ILLEGAL AC.
 =====
 TEST THAT THE SPECIAL FDST FLOW USED BY ABS, NEG, AND TST
 WILL TRAP IF AN ILLEGAL MODE 0 AC IS USED.
- 9.17 TEST 17 -- SPECIAL FDST MODE 0, USING NEGD.
 =====
 TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
 INSTRUCTIONS, USING NEGD, MODE 0.
 EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
 AND AGAIN WITH E(FDST) NON-ZERO.
- 9.20 TEST 20 -- SPECIAL FDST MODE 1, USING NEGD.
 =====
 TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
 INSTRUCTIONS, USING NEGD, MODE 1.
 EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
 AND AGAIN WITH E(FDST) NON-ZERO.

- 9.21 TEST 21 -- SPECIAL FDST MODE 2, USING ABSD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING ABSD, MODE 2.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.
- 9.22 TEST 22 -- SPECIAL FDST MODE 4, USING ABSD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING ABSD, MODE 4.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.
- 9.23 TEST 23 -- SPECIAL FDST MODE 3, USING NEGD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING NEGD, MODE 3.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.
- 9.24 TEST 24 -- SPECIAL FDST MODE 5, USING NEGD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING NEGD, MODE 5.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.
- 9.25 TEST 25 -- SPECIAL FDST MODE 6, USING ABSD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING ABSD, MODE 6.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.
- 9.26 TEST 26 -- SPECIAL FDST MODE 7, USING ABSD.
=====
- TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST INSTRUCTIONS, USING ABSD, MODE 7.
EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
AND AGAIN WITH E(FDST) NON-ZERO.

- 9.27 TEST 27 -- SPECIAL FDST MODE 6 WITH R7, USING NEGD.
=====
- TEST SPECIAL FDST FLOW, MODE 6, WITH GR7 (PC RELATIVE),
USING THE NEGD INSTRUCTION.
- 9.30 TEST 30 -- SPECIAL FDST MODE 7 WITH R7, USING ABSD.
=====
- TEST SPECIAL FDST FLOW, MODE 7, WITH GR7 (PC RELATIVE DEFERRED),
USING THE ABSD INSTRUCTION.
- 9.31 TEST 31 -- SPECIAL FDST MODE 2 WITH R7, USING NEGD.
=====
- TEST SPECIAL FDST FLOW, MODE 2, WITH GR7 (PC IMMEDIATE),
USING THE NEGD INSTRUCTION.
- 9.32 TEST 32 -- NEGD, ABSD, AND TSTD (FDST MODE 1).
=====
- TEST NEGD, ABSD, AND TSTD, WITH VARIOUS OPERANDS.
- 9.33 TEST 33 -- LDFPS (SRC MODE 1).
=====
- TEST SRC MODE 1, USING THE LDFPS INSTRUCTION.
- 9.34 TEST 34 -- LDFPS (SRC MODE 2).
=====
- TEST SRC MODE 2, USING THE LDFPS INSTRUCTION.
- 9.35 TEST 35 -- LDFPS (SRC MODE 4).
=====
- TEST SRC MODE 4, USING THE LDFPS INSTRUCTION.
- 9.36 TEST 36 -- LDFPS (SRC MODE 3).
=====
- TEST SRC MODE 3, USING THE LDFPS INSTRUCTION.
- 9.37 TEST 37 -- LDFPS (SRC MODE 5).
=====
- TEST SRC MODE 5, USING THE LDFPS INSTRUCTION.

- 9.40 TEST 40 -- LDFPS (SRC MODE 6).
=====
- TEST SRC MODE 6, USING THE LDFPS INSTRUCTION.
- 9.41 TEST 41 -- LDFPS (SRC MODE 7).
=====
- TEST SRC MODE 7, USING THE LDFPS INSTRUCTION.
- 9.42 TEST 42 -- LDCLD (SRC MODE 2 WITH R7).
=====
- TEST SRC MODE 2(IMMEDIATE), USING THE LDCLD INSTRUCTION.
- 9.43 TEST 43 -- LDCLD (SRC MODE 2).
=====
- TEST SRC MODE 2(AUTO-INCR), USING THE LDCLD INSTRUCTION.
- 9.44 TEST 44 -- LDCIF AND LDCLF (SRC MODE 1).
=====
- TEST THE LDCIF/LDCLF INSTRUCTION, USING VARIOUS OPERANDS.
CONVERT FROM 'I' OR 'L' TO 'F' (SINGLE-PRECISION FLOATING).
- 9.45 TEST 45 -- LDCID AND LDCLD (SRC MODE 1).
=====
- TEST THE LDCID/LDCLD INSTRUCTION, USING VARIOUS OPERANDS.
CONVERT FROM 'I' OR 'L' TO 'D' (DOUBLE-PRECISION FLOATING).
- 9.46 TEST 46 -- LDEXP (SRC MODES 0 AND 1).
=====
- TEST THE LDEXP INSTRUCTION WITH A VARIETY OF OPERANDS.
BOTH SRC MODES 0 AND 1 ARE TESTED FOR EACH OPERAND SET.
- 9.47 TEST 47 -- STFPS (DST MODE 1).
=====
- TEST DST MODE 1, USING THE STFPS INSTRUCTION.
- 9.50 TEST 50 -- STFPS (DST MODE 2).
=====
- TEST DST MODE 2, USING THE STFPS INSTRUCTION.

- 9.51 TEST 51 -- STFPS (DST MODE 4).
=====
- TEST DST MODE 4, USING THE STFPS INSTRUCTION.
- 9.52 TEST 52 -- STFPS (DST MODE 3).
=====
- TEST DST MODE 3, USING THE STFPS INSTRUCTION.
- 9.53 TEST 53 -- STFPS (DST MODE 5).
=====
- TEST DST MODE 5, USING THE STFPS INSTRUCTION.
- 9.54 TEST 54 -- STFPS (DST MODE 6).
=====
- TEST DST MODE 6, USING THE STFPS INSTRUCTION.
- 9.55 TEST 55 -- STFPS (DST MODE 7).
=====
- TEST DST MODE 7, USING THE STFPS INSTRUCTION.
- 9.56 TEST 56 -- STCDL (DST MODES 2 AND 4).
=====
- TEST DST MODES 2 AND 4, USING THE STCDL INSTRUCTION.
- 9.57 TEST 57 -- STCDI/STCDL AND STCFI/STCFL (DST MODE 1).
=====
- TEST THE 'STORE CONVERTED' INSTRUCTIONS STCDI, STCDL
STCFI, AND STCFL.
- 9.60 TEST 60 -- STEXP (DST MODES 0 AND 1).
=====
- TEST THE STEXP INSTRUCTION WITH A VARIETY OF OPERANDS.
BOTH DST MODES 0 AND 1 ARE TESTED FOR EACH OPERAND SET.
- 9.61 TEST 61 -- STST (DST MODE 1).
=====
- TEST THE STST INSTRUCTION USING FDST MODE 1.
VERIFY THAT THE RETURNED FEC AND FEA ARE CORRECT.

9.62 TEST 62 -- FPF11 INTERRUPTABILITY.

=====

THIS TEST IS INCLUDED IN CASE THE FPF11 PRESENTS INTERRUPT LATENCY PROBLEMS. AT THE PRESENT TIME, FPF11 INSTRUCTIONS ARE NOT INTERRUPTABLE. HOPEFULLY, THIS WON'T BE A PROBLEM. IF IT TURNS OUT THAT LATENCY IS EXCESSIVE -- THE MICROCODE MAY HAVE TO BE TWEAKED TO PROVIDE INTERRUPTABILITY. THIS TEST WILL EXECUTE ADDD, SUBD, MULD, DIVD, AND MODD OPCODES, ATTEMPT TO INTERRUPT (ABORT) THEM VIA TTY INTERRUPT, AND REPORT WHETHER OR NOT THE INSTRUCTION WAS IN FACT INTERRUPTED. NO SPECIAL EQUIPMENT (OTHER THAN THE CONSOLE TTY) IS REQUIRED.

*** NOTE ***

THE TEST IS NOT INCLUDED IN THE NORMAL TEST SEQUENCE. IF YOU WANT TO RUN IT -- CHANGE THE SWR TO 000462 (LOOP ON TEST 62).

10.0 PROGRAM LISTING

=====

THE PROGRAM LISTING FOLLOWS:

CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2 MACY11 30G(1063) 12-FEB-81 11:04
 CJFPBA.P11 12-FEB-81 10:27 TABLE OF CONTENTS

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 2 TESTS
210	
216	T1 STF -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
254	T2 STF -- FDST MODE 1 (INDIRECT)
306	T3 STF AND STD -- FDST MODE 2 (AUTO-INCREMENT)
389	T4 STD -- FDST MODE 2, WITH GR7 (PC IMMEDIATE)
442	T5 STD -- FDST MODE 4 (AUTO-DECREMENT)
493	T6 STD -- FDST MODE 3 (AUTO-INCR DEFERRED)
547	T7 STD -- FDST MODE 5 (AUTO-DECR DEFERRED)
601	T10 STD -- FDST MODE 6 (INDEX)
652	T11 STD -- FDST MODE 7 (INDEX DEFERRED)
706	T12 STCFD AND STCDF -- FDST MODE 1
870	T13 STCFD -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
901	T14 CLRD AND CLRD -- FDST MODE 1
950	T15 CLRD -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
980	T16 SPECIAL FDST FLOW, USING NEGD MODE 0 WITH ILLEGAL AC7
1012	T17 SPECIAL FDST FLOW, USING NEGD MODE 0
1059	T20 SPECIAL FDST FLOW, USING NEGD MODE 1
1124	T21 SPECIAL FDST FLOW, USING ABSD MODE 2
1190	T22 SPECIAL FDST FLOW, USING ABSD MODE 4
1256	T23 SPECIAL FDST FLOW, USING NEGD MODE 3
1325	T24 SPECIAL FDST FLOW, USING NEGD MODE 5
1394	T25 SPECIAL FDST FLOW, USING ABSD MODE 6
1459	T26 SPECIAL FDST FLOW, USING ABSD MODE 7
1532	T27 SPECIAL FDST FLOW, USING NEGD MODE 6 WITH GR7
1594	T30 SPECIAL FDST FLOW, USING ABSD MODE 7 WITH GR7
1658	T31 SPECIAL FDST FLOW, USING NEGD MODE 2 WITH GR7
1712	T32 NEGD, ABSD, AND TSTD -- FDST MODE 1
1876	T33 LDFPS -- SRC MODE 1
1915	T34 LDFPS -- SRC MODE 2
1954	T35 LDFPS -- SRC MODE 4
1993	T36 LDFPS -- SRC MODE 3
2034	T37 LDFPS -- SRC MODE 5
2075	T40 LDFPS -- SRC MODE 6
2114	T41 LDFPS -- SRC MODE 7
2155	T42 LDCLD -- SRC MODE 2 WITH GR7
2195	T43 LDCLD -- SRC MODE 2
2226	T44 LDCIF AND LDCLF -- SRC MODE 1

2429	T45	LDCID AND LDCLD -- SRC MODE 1
2562	T46	LDEXP -- SRC MODE 0 AND 1
2800	T47	STFPS -- DST MODE 1
2844	T50	STFPS -- DST MODE 2
2889	T51	STFPS -- DST MODE 4

2934	T52	STFPS -- DST MODE 3
2980	T53	STFPS -- DST MODE 5
3025	T54	STFPS -- DST MODE 6
3069	T55	STFPS -- DST MODE 7
3119	T56	STCDL -- DST MODES 2 AND 4
3167	T57	STCDI/STCDL AND STCFI/STCFL -- DST MODE 1
3429	T60	STEXP -- DST MODE 0 AND 1
3547	T61	STST -- DST MODE 1
3598	T62	INTERRUPTABILITY TEST
3698		
3699		END OF PASS ROUTINE
3701		SCOPE HANDLER ROUTINE
3731		TYPE ROUTINE
3744		GET VALUE FOR SOFTWARE SWITCH REGISTER
3748		BINARY TO OCTAL (ASCII) AND TYPE
3750		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3762		APT COMMUNICATIONS ROUTINE
3764		TTY INPUT ROUTINE
3766		TRAP DECODER
(3)		TRAP TABLE
3794		POWER DOWN AND UP ROUTINES
3796		ERROR HANDLER ROUTINE
3798		ERROR TYPE OUT ROUTINE
3901		ASCII TEXT AND ERROR MESSAGES
4462		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)
(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)
(1)

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

.ENABL ABS,AMA

.TITLE CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
;*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-11-DZQAC-C4), 31 JULY 1980.
;*
(1)      000001 $TN=1
(1)      160000 $SWR=160000 ;:HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
;*
;* THESE PROGRAMS WERE ADAPTED FROM THE ORIGINAL 11/34 FPF11-A
;* DIAGNOSTICS WRITTEN BY ANTHONY S. VEZZA IN SEPTEMBER '76.
;*
;* FPF PART 1 INCLUDES ALL TESTS FOUND IN FPF11-A, PARTS 1 AND 2.
;* FPF PART 2 INCLUDES ALL TESTS FOUND IN FPF11-A, PART 3.
;* ERROR ANALYSIS AND REPORTING HAS BEEN REWORKED TO ACCOMODATE
;* THE 11/23 FPF11 MICROCODE AND MAINTENANCE PHILOSOPHY.
;*
;*
;*                                     G.P. JAN '80
;*
(1)      000114 PRGSIZ= ^H<LASTAD> ; PROGRAM SIZE IN 1/8 K UNITS (OCTAL).
;*****
;*****
.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)      001100 STACK= 1100
(1)      .EQUIV EMT,ERROR ;:BASIC DEFINITION OF ERROR CALL
(1)      .EQUIV IOT,SCOPE ;:BASIC DEFINITION OF SCOPE CALL
;*MISCELLANEOUS DEFINITIONS
(1)      000011 HT= 11 ;:CODE FOR HORIZONTAL TAB
(1)      000012 LF= 12 ;:CODE FOR LINE FEED
(1)      000015 CR= 15 ;:CODE FOR CARRIAGE RETURN
(1)      000200 CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
(1)      177776 PS= 177776 ;:PROCESSOR STATUS WORD
(1)      .EQUIV PS,PSW
(1)      177774 STKLMT= 177774 ;:STACK LIMIT REGISTER
(1)      177772 PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
(1)      177570 DSWR= 177570 ;:HARDWARE SWITCH REGISTER
(1)      177570 DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
```

```
(1)
(1)
(1) 000000 R0= X0      ;;GENERAL REGISTER
(1) 000001 R1= X1      ;;GENERAL REGISTER
(1) 000002 R2= X2      ;;GENERAL REGISTER
(1) 000003 R3= X3      ;;GENERAL REGISTER
(1) 000004 R4= X4      ;;GENERAL REGISTER
(1) 000005 R5= X5      ;;GENERAL REGISTER
(1) 000006 R6= X6      ;;GENERAL REGISTER
(1) 000007 R7= X7      ;;GENERAL REGISTER
(1) 000006 SP= X6      ;;STACK POINTER
(1) 000007 PC= X7      ;;PROGRAM COUNTER
```

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

```
(1)
(1)
(1) 100000 SW15= 100000 ;;'SWITCH REGISTER' SWITCH DEFINITIONS
(1) 040000 SW14= 40000
(1) 020000 SW13= 20000
(1) 010000 SW12= 10000
(1) 004000 SW11= 4000
(1) 002000 SW10= 2000
(1) 001000 SW09= 1000
(1) 000400 SW08= 400
(1) 000200 SW07= 200
(1) 000100 SW06= 100
(1) 000040 SW05= 40
(1) 000020 SW04= 20
(1) 000010 SW03= 10
(1) 000004 SW02= 4
(1) 000002 SW01= 2
(1) 000001 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 BIT15= 100000 ;;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1) 040000 BIT14= 40000
(1) 020000 BIT13= 20000
(1) 010000 BIT12= 10000
```

BASIC DEFINITIONS

```

(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
80
81 ;
82 ; FPF-11 DEFINITIONS.
83 ;
84 FPVEC= 244 ; THE STANDARD FP VECTOR...
85 AC0= %0 ;...AND ACCUMULATORS.
86 AC1= %1
87 AC2= %2
88 AC3= %3
89 AC4= %4
90 AC5= %5
91 AC6= %6
92 AC7= %7
93 ;
94 ; MISCELLANEOUS DEFINITIONS.
95 ;
96 TAB= HT
97 SKP1 BR+1
98 SKP2= BR+2
99 SKP3= BR+3
  
```

```

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  DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
(1) 000176 000000  SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
(1)      .SBTTL STARTING ADDRESS(ES)
(1) 000200 000137 003312 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>
105      ;*      7      ENABLE ANALYZER (RESERVED)
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 027100      $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) 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	\$STSM: .WORD	5	:: RUN TIME 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:      .;START OF COMMON TAGS
(1)      001102      000      $TSTNM: .WORD 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)                      ;.SBTTL APT MAILBOX-ETABLE
(2)                      ;:*****
(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)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2) 001264 000        $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS, M.S. BYTE
(2) 001265 000        $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE, BLK#1
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2)                      ;*
(2) 001266 000000      $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS, BLK#1
(2)                      ;*
(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#2, BUS 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) : * 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 000200 LASTEM= 200 ; 200 ERRORS TOTAL.

120	001312	033443	044176	044514	.WORD	EM1, DH1, DT1, DF1	: ERROR ITEM 1
(3)	001322	033516	044243	044534	.WORD	EM2, DH2, DT2, DF2	: ERROR ITEM 2
(3)	001332	000000	000000	000000	.WORD	0,0,0,0	
(3)	001342	033572	044243	044556	.WORD	EM4, DH4, DT4, DF4	: ERROR ITEM 4
(3)	001352	033642	044243	044604	.WORD	EM5, DH5, DT5, DF5	: ERROR ITEM 5
(3)	001362	033705	044243	044604	.WORD	EM6, DH6, DT6, DF6	: ERROR ITEM 6
(3)	001372	033761	044243	044556	.WORD	EM7, DH7, DT7, DF7	: ERROR ITEM 7
(3)	001402	034031	044243	044604	.WORD	EM10, DH10, DT10, DF10	: ERROR ITEM 10
(3)	001412	034074	044243	044604	.WORD	EM11, DH11, DT11, DF11	: ERROR ITEM 11
(3)	001422	034140	044274	044514	.WORD	EM12, DH12, DT12, DF12	: ERROR ITEM 12
(3)	001432	034211	044344	044632	.WORD	EM13, DH13, DT13, DF13	: ERROR ITEM 13
(3)	001442	034245	044243	044556	.WORD	EM14, DH14, DT14, DF14	: ERROR ITEM 14
(3)	001452	034316	044243	044604	.WORD	EM15, DH15, DT15, DF15	: ERROR ITEM 15
(3)	001462	034362	044344	044632	.WORD	EM16, DH16, DT16, DF16	: ERROR ITEM 16
(3)	001472	034416	044243	044556	.WORD	EM17, DH17, DT17, DF17	: ERROR ITEM 17
(3)	001502	034470	044243	044604	.WORD	EM20, DH20, DT20, DF20	: ERROR ITEM 20
(3)	001512	034535	044344	044632	.WORD	EM21, DH21, DT21, DF21	: ERROR ITEM 21
(3)	001522	034572	044243	044556	.WORD	EM22, DH22, DT22, DF22	: ERROR ITEM 22
(3)	001532	034644	044243	044604	.WORD	EM23, DH23, DT23, DF23	: ERROR ITEM 23
(3)	001542	034711	044344	044632	.WORD	EM24, DH24, DT24, DF24	: ERROR ITEM 24
(3)	001552	034746	044243	044556	.WORD	EM25, DH25, DT25, DF25	: ERROR ITEM 25
(3)	001562	035021	044243	044604	.WORD	EM26, DH26, DT26, DF26	: ERROR ITEM 26
(3)	001572	035067	044344	044632	.WORD	EM27, DH27, DT27, DF27	: ERROR ITEM 27
(3)	001602	035125	044243	044556	.WORD	EM30, DH30, DT30, DF30	: ERROR ITEM 30
(3)	001612	035201	044243	044604	.WORD	EM31, DH31, DT31, DF31	: ERROR ITEM 31
(3)	001622	035250	044344	044632	.WORD	EM32, DH32, DT32, DF32	: ERROR ITEM 32
(3)	001632	000000	000000	000000	.WORD	0,0,0,0	
(3)	001642	035307	044243	044644	.WORD	EM34, DH34, DT34, DF34	: ERROR ITEM 34
(3)	001652	035375	044243	044534	.WORD	EM35, DH35, DT35, DF35	: ERROR ITEM 35
(3)	001662	035466	044374	044702	.WORD	EM36, DH36, DT36, DF36	: ERROR ITEM 36
(3)	001672	035537	044243	044534	.WORD	EM37, DH37, DT37, DF37	: ERROR ITEM 37
(3)	001702	035623	044243	044534	.WORD	EM40, DH40, DT40, DF40	: ERROR ITEM 40
(3)	001712	035707	044374	044732	.WORD	EM41, DH41, DT41, DF41	: ERROR ITEM 41
(3)	001722	035757	044243	044556	.WORD	EM42, DH42, DT42, DF42	: ERROR ITEM 42

(3)	001732	036024	044374	044766	.WORD	EM43,	DH43,	DT43,	DF43	:	ERROR	ITEM	43
(3)	001742	036075	044344	044632	.WORD	EM44,	DH44,	DT44,	DF44	:	ERROR	ITEM	44
(3)	001752	036125	044243	044556	.WORD	EM45,	DH45,	DT45,	DF45	:	ERROR	ITEM	45
(3)	001762	036173	044374	044766	.WORD	EM46,	DH46,	DT46,	DF46	:	ERROR	ITEM	46
(3)	001772	036245	044344	044632	.WORD	EM47,	DH47,	DT47,	DF47	:	ERROR	ITEM	47
(3)	002002	036276	044243	044556	.WORD	EM50,	DH50,	DT50,	DF50	:	ERROR	ITEM	50
(3)	002012	036344	044374	044766	.WORD	EM51,	DH51,	DT51,	DF51	:	ERROR	ITEM	51
(3)	002022	036416	044344	044632	.WORD	EM52,	DH52,	DT52,	DF52	:	ERROR	ITEM	52
(3)	002032	036447	044243	044556	.WORD	EM53,	DH53,	DT53,	DF53	:	ERROR	ITEM	53
(3)	002042	036516	044374	044766	.WORD	EM54,	DH54,	DT54,	DF54	:	ERROR	ITEM	54
(3)	002052	036571	044344	044632	.WORD	EM55,	DH55,	DT55,	DF55	:	ERROR	ITEM	55
(3)	002062	036623	044243	044556	.WORD	EM56,	DH56,	DT56,	DF56	:	ERROR	ITEM	56
(3)	002072	036672	044374	044766	.WORD	EM57,	DH57,	DT57,	DF57	:	ERROR	ITEM	57
(3)	002102	036745	044344	044632	.WORD	EM60,	DH60,	DT60,	DF60	:	ERROR	ITEM	60
(3)	002112	036777	044243	044556	.WORD	EM61,	DH61,	DT61,	DF61	:	ERROR	ITEM	61
(3)	002122	037047	044374	044766	.WORD	EM62,	DH62,	DT62,	DF62	:	ERROR	ITEM	62
(3)	002132	037123	044344	044632	.WORD	EM63,	DH63,	DT63,	DF63	:	ERROR	ITEM	63
(3)	002142	037156	044243	044556	.WORD	EM64,	DH64,	DT64,	DF64	:	ERROR	ITEM	64
(3)	002152	037227	044374	044766	.WORD	EM65,	DH65,	DT65,	DF65	:	ERROR	ITEM	65
(3)	002162	037304	044344	044632	.WORD	EM66,	DH66,	DT66,	DF66	:	ERROR	ITEM	66
(3)	002172	037340	044374	044766	.WORD	EM67,	DH67,	DT67,	DF67	:	ERROR	ITEM	67
(3)	002202	037427	044344	044632	.WORD	EM70,	DH70,	DT70,	DF70	:	ERROR	ITEM	70
(3)	002212	037475	044374	044766	.WORD	EM71,	DH71,	DT71,	DF71	:	ERROR	ITEM	71
(3)	002222	037564	044344	044632	.WORD	EM72,	DH72,	DT72,	DF72	:	ERROR	ITEM	72
(3)	002232	037632	044374	044766	.WORD	EM73,	DH73,	DT73,	DF73	:	ERROR	ITEM	73
(3)	002242	037704	044274	044514	.WORD	EM74,	DH74,	DT74,	DF74	:	ERROR	ITEM	74
(3)	002252	037752	044344	044632	.WORD	EM75,	DH75,	DT75,	DF75	:	ERROR	ITEM	75
(3)	002262	040003	044243	045022	.WORD	EM76,	DH76,	DT76,	DF76	:	ERROR	ITEM	76
(3)	002272	000000	000000	000000	.WORD	0,0,0,0							
(3)	002302	040057	044243	044556	.WORD	EM100,	DH100,	DT100,	DF100	:	ERROR	ITEM	100
(3)	002312	040125	044374	044514	.WORD	EM101,	DH101,	DT101,	DF101	:	ERROR	ITEM	101
(3)	002322	040165	044344	044632	.WORD	EM102,	DH102,	DT102,	DF102	:	ERROR	ITEM	102
(3)	002332	040216	044243	044556	.WORD	EM103,	DH103,	DT103,	DF103	:	ERROR	ITEM	103
(3)	002342	040265	044374	044514	.WORD	EM104,	DH104,	DT104,	DF104	:	ERROR	ITEM	104
(3)	002352	040326	044344	044632	.WORD	EM105,	DH105,	DT105,	DF105	:	ERROR	ITEM	105
(3)	002362	040360	044243	044556	.WORD	EM106,	DH106,	DT106,	DF106	:	ERROR	ITEM	106
(3)	002372	040427	044374	044514	.WORD	EM107,	DH107,	DT107,	DF107	:	ERROR	ITEM	107
(3)	002402	040470	044344	044632	.WORD	EM110,	DH110,	DT110,	DF110	:	ERROR	ITEM	110
(3)	002412	040522	044243	044556	.WORD	EM111,	DH111,	DT111,	DF111	:	ERROR	ITEM	111
(3)	002422	040572	044374	044514	.WORD	EM112,	DH112,	DT112,	DF112	:	ERROR	ITEM	112
(3)	002432	040634	044344	044632	.WORD	EM113,	DH113,	DT113,	DF113	:	ERROR	ITEM	113
(3)	002442	040667	044243	044556	.WORD	EM114,	DH114,	DT114,	DF114	:	ERROR	ITEM	114
(3)	002452	040737	044374	044514	.WORD	EM115,	DH115,	DT115,	DF115	:	ERROR	ITEM	115
(3)	002462	041001	044344	044632	.WORD	EM116,	DH116,	DT116,	DF116	:	ERROR	ITEM	116
(3)	002472	041034	044243	044556	.WORD	EM117,	DH117,	DT117,	DF117	:	ERROR	ITEM	117
(3)	002502	041105	044374	044514	.WORD	EM120,	DH120,	DT120,	DF120	:	ERROR	ITEM	120
(3)	002512	041150	044344	044632	.WORD	EM121,	DH121,	DT121,	DF121	:	ERROR	ITEM	121
(3)	002522	041204	044243	044556	.WORD	EM122,	DH122,	DT122,	DF122	:	ERROR	ITEM	122
(3)	002532	041256	044374	044514	.WORD	EM123,	DH123,	DT123,	DF123	:	ERROR	ITEM	123
(3)	002542	041322	044344	044632	.WORD	EM124,	DH124,	DT124,	DF124	:	ERROR	ITEM	124
(3)	002552	041357	044274	044514	.WORD	EM125,	DH125,	DT125,	DF125	:	ERROR	ITEM	125
(3)	002562	041423	044344	044632	.WORD	EM126,	DH126,	DT126,	DF126	:	ERROR	ITEM	126
(3)	002572	041461	044243	044556	.WORD	EM127,	DH127,	DT127,	DF127	:	ERROR	ITEM	127
(3)	002602	041534	044374	045060	.WORD	EM130,	DH130,	DT130,	DF130	:	ERROR	ITEM	130
(3)	002612	041612	044374	045060	.WORD	EM131,	DH131,	DT131,	DF131	:	ERROR	ITEM	131
(3)	002622	041670	044243	045114	.WORD	EM132,	DH132,	DT132,	DF132	:	ERROR	ITEM	132

(3)	002632	041751	044243	044556	.WORD	EM133,	DH133,	DT133,	DF133	:	ERROR	ITEM	133
(3)	002642	042017	044176	045156	.WORD	EM134,	DH134,	DT134,	DF134	:	ERROR	ITEM	134
(3)	002652	042067	044344	044632	.WORD	EM135,	DH135,	DT135,	DF135	:	ERROR	ITEM	135
(3)	002662	042120	044243	044556	.WORD	EM136,	DH136,	DT136,	DF136	:	ERROR	ITEM	136
(3)	002672	042167	044176	045156	.WORD	EM137,	DH137,	DT137,	DF137	:	ERROR	ITEM	137
(3)	002702	042240	044344	044632	.WORD	EM140,	DH140,	DT140,	DF140	:	ERROR	ITEM	140
(3)	002712	042272	044243	044556	.WORD	EM141,	DH141,	DT141,	DF141	:	ERROR	ITEM	141
(3)	002722	042341	044176	045156	.WORD	EM142,	DH142,	DT142,	DF142	:	ERROR	ITEM	142
(3)	002732	042412	044344	044632	.WORD	EM143,	DH143,	DT143,	DF143	:	ERROR	ITEM	143
(3)	002742	042444	044243	044556	.WORD	EM144,	DH144,	DT144,	DF144	:	ERROR	ITEM	144
(3)	002752	042514	044176	045156	.WORD	EM145,	DH145,	DT145,	DF145	:	ERROR	ITEM	145
(3)	002762	042566	044344	044632	.WORD	EM146,	DH146,	DT146,	DF146	:	ERROR	ITEM	146
(3)	002772	042621	044243	044556	.WORD	EM147,	DH147,	DT147,	DF147	:	ERROR	ITEM	147
(3)	003002	042671	044176	045156	.WORD	EM150,	DH150,	DT150,	DF150	:	ERROR	ITEM	150
(3)	003012	042743	044344	044632	.WORD	EM151,	DH151,	DT151,	DF151	:	ERROR	ITEM	151
(3)	003022	042776	044243	044556	.WORD	EM152,	DH152,	DT152,	DF152	:	ERROR	ITEM	152
(3)	003032	043047	044176	045156	.WORD	EM153,	DH153,	DT153,	DF153	:	ERROR	ITEM	153
(3)	003042	043122	044344	044632	.WORD	EM154,	DH154,	DT154,	DF154	:	ERROR	ITEM	154
(3)	003052	043156	044243	044556	.WORD	EM155,	DH155,	DT155,	DF155	:	ERROR	ITEM	155
(3)	003062	043230	044176	045156	.WORD	EM156,	DH156,	DT156,	DF156	:	ERROR	ITEM	156
(3)	003072	043304	044344	044632	.WORD	EM157,	DH157,	DT157,	DF157	:	ERROR	ITEM	157
(3)	003102	043341	044243	044556	.WORD	EM160,	DH160,	DT160,	DF160	:	ERROR	ITEM	160
(3)	003112	043414	044243	044556	.WORD	EM161,	DH161,	DT161,	DF161	:	ERROR	ITEM	161
(3)	003122	043467	044243	045204	.WORD	EM162,	DH162,	DT162,	DF162	:	ERROR	ITEM	162
(3)	003132	043555	044374	045242	.WORD	EM163,	DH163,	DT163,	DF163	:	ERROR	ITEM	163
(3)	003142	043644	044374	045276	.WORD	EM164,	DH164,	DT164,	DF164	:	ERROR	ITEM	164
(3)	003152	043720	044446	044514	.WORD	EM165,	DH165,	DT165,	DF165	:	ERROR	ITEM	165
(3)	003162	044003	044446	044514	.WORD	EM166,	DH166,	DT166,	DF166	:	ERROR	ITEM	166
(3)	003172	000000	000000	000000	.WORD	0,0,0,0							
(3)	003202	000000	000000	000000	.WORD	0,0,0,0							
(3)	003212	000000	000000	000000	.WORD	0,0,0,0							
(3)	003222	000000	000000	000000	.WORD	0,0,0,0							
(3)	003232	000000	000000	000000	.WORD	0,0,0,0							
(3)	003242	000000	000000	000000	.WORD	0,0,0,0							
(3)	003252	044054	044176	044514	.WORD	EM175,	DH175,	DT175,	DF175	:	ERROR	ITEM	175
(3)	003262	044110	044344	044632	.WORD	EM176,	DH176,	DT176,	DF176	:	ERROR	ITEM	176
(3)	003272	044142	044344	044632	.WORD	EM177,	DH177,	DT177,	DF177	:	ERROR	ITEM	177
(3)	003302	000000	000000	000000	.WORD	0,0,0,0							

```

122      ;
123      ; INITIAL START/RESTART HERE.
124      ;
126      003312
127      003312 000240
128      ;
(1)      ; START:RESTART:
(1)      003314 012706 001100      NOP
(1)      003320 005026              .SBTTL INITIALIZE THE COMMON TAGS
(1)      003322 022706 001140      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1)      003326 001374              MOV     # $CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(1)      003330 012706 001100      CLR     (R6)+           ;;CLEAR MEMORY LOCATION
(1)      ;                               CMP     #SWR,R6 ;;DONE?
(1)      ;                               BNE     -6              ;;LOOP BACK IF NO
(1)      ;                               MOV     #STACK,SP      ;;SETUP THE STACK POINTER
(1)      ;                               ;;INITIALIZE A FEW VECTORS
(1)      003334 012737 027120 000020 MOV     # $SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1)      003342 012737 000340 000022 MOV     #340,@#IOTVEC+2 ;;LEVEL 7
(1)      003350 012737 032200 000030 MOV     # $ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1)      003356 012737 000340 000032 MOV     #340,@#EMTVEC+2 ;;LEVEL 7
(1)      003364 012737 031656 000034 MOV     # $TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1)      003372 012737 000340 000036 MOV     #340,@#TRAPVEC+2;LEVEL 7
(1)      003400 012737 032026 000024 MOV     # $PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
(1)      003406 012737 000340 000026 MOV     #340,@#PWRVEC+2 ;;LEVEL 7
(1)      003414 013737 026744 026736 MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
(1)      003422 005037 001220      CLR     $TIMES         ;;INITIALIZE NUMBER OF ITERATIONS
(1)      003426 005037 001222      CLR     $ESCAPE       ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1)      003432 112737 000001 001115 MOV     #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
(1)      003440 012737 003440 001106 MOV     #,$SLPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1)      003446 012737 003446 001110 MOV     #,$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)      003454 013746 000004      MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
(2)      003460 012737 003514 000004 MOV     #64$,@#ERRVEC  ;;SET UP ERROR VECTOR
(2)      003466 012737 177570 001140 MOV     #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
(2)      003474 012737 177570 001142 MOV     #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2)      003502 022777 177777 175430 CMP     #-1,@SWR       ;;TRY TO REFERENCE HARDWARE SWR
(2)      003510 001012      BNE     66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)      ;                               ;;AND THE HARDWARE SWR IS NOT - -1
(2)      003512 000403      BR     65$           ;;BRANCH IF NO TIMEOUT
(2)      003514 012716 003522      64$: MOV     #65$,(SP)   ;;SET UP FOR TRAP RETURN
(2)      003520 000002      RTI
(2)      003522 012737 000176 001140 65$: MOV     #SWREG,SWR    ;;POINT TO SOFTWARE SWR
(2)      003530 012737 000174 001142 MOV     #DISPREG,DISPLAY
(2)      003536 012637 000004      66$: MOV     (SP)+,@#ERRVEC ;;RESTORE ERROR VECIOR
(1)      ;
(2)      003542 005037 001242      CLR     $PASS         ;;CLEAR PASS COUNT
(2)      003546 132737 000200 001255 BITB    #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2)      003554 001403      BEQ    67$           ;;YES,USE NON-APT SWITCH
(2)      003556 012737 001256 001140 MOV     # $SWREG,SWR   ;;NO,USE APT SWITCH REGISTER
(2)      003564      67$:
    
```

```

130
131
132
133
134 003564 005227 177777
135 003570 001032
136 003572 023727 000042 027100
137 003600 001426
138 003602 104401 003610
(1) 003606 000423
(1)
(1) 003656
139 003656
(1)
(1) 003656 005737 000042
(1) 003662 001012
(1) 003667 123727 001254 000001
(1) 003672 001406
(1) 003674 023727 001140 000176
(1) 003702 001005
(1) 003704 104406
(1) 003706 000403
(1) 003710 112737 000001 001134
(1) 003716
140
141
142
143 003716 012706 001100
144 003722 012737 003750 000244
145 003730 012737 004004 000010
146 003736 012737 003774 000004
147 003744 000137 004126

; 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, # $ENDAD ; ACT-11 ??
; BEQ 1$ ; YES
; TYPE ,69$ ; TYPE ASCIZ STRING
; BR 68$ ; GET OVER THE ASCIZ
;:69$: .ASCIZ <CRLF>'CJFPBA -- FPF11 DIAGNOSTIC, PART 2'<CRLF>
68$:
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$
(1) 003710 112737 000001 001134 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 TST ; 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      TRP244: MOV      (SP), $TMP2      ; GET PC OF TRAP.
156              STFPS   R0              ; GET FPS
157              MOV      R0, $TMP3
158              STST    R0              ; GET FEC
159              MOV      R0, $TMP4
160              ERROR   175
161              BR      1$              ; COMMON EXIT.
162
163      TRP04: MOV      (SP), $TMP2      ; GET PC OF TRAP.
164              ERROR   176
165              BR      1$
166
167      TRP10: MOV      (SP), $TMP2      ; GET PC OF TRAP.
168              ERROR   177
169
170      1$:  CMP      (SP)+, (SP)+      ; COMMON EXIT, FIX STACK...
171              TYPE    ,ABORT         ; ...AND TELL THE MAN.
172              CMPB    $TSTNM, #LASTST ; ON THE LAST TEST ??
173              BNE     2$              ; NO.
174              JMP     $EOP            ; YES, END-PASS.
175      2$:  MOVB    $TSTNM, R0         ; GET TEST NUM...
176              ASL     R0              ; ...SHIFT TO WORD INDEX.
177              MOV     $$W08TBL(R0), R1 ; GET NEXT TEST ADDRESS...
178              JMP     -2(R1)          ; ...AND GO THERE.
179      .DSABL  LSB
    
```

```

181
182
183
184
185
186
187
188
189 004052 012737 000002 004122 CHECK2: .ENABL  LSB
190 004060 000403 SKP3
191 004062 012737 000004 004122 CHECK4: MOV #4,3$ ; 64 BIT COMPARE.
192 004070 010046 MOV R0,-(SP) ; SAVE REGISTERS.
193 004072 010146 MOV R1,-(SP)
194 004074 012500 MOV (R5)+,R0 ; SET OPERAND ADDRESSES.
195 004076 012501 MOV (R5)+,R1
196 004100 000240 NOP
197 004102 022021 1$: CMP (R0)+,(R1)+ ; COMPARE.
198 004104 001003 BNE 2$ ; EXIT IF NOT EQUAL.
199 004106 005337 004122 DEC 3$
200 004112 001373 BNE 1$
201 004114 012601 2$: MOV (SP)+,R1 ; RESTORE REGISTERS.
202 004116 012600 MOV (SP)+,R0
203 004120 005727 TST (PC)+
204 004122 000000 3$: 0 ; 0 IF EQUAL, NZ OTHERWISE.
205 004124 000205 RTS R5
206 .DSABL LSB
  
```

```
208 .SBTTL
209 .SBTTL          FPF PART 2 TESTS
210 .SBTTL
216 :*****
(3) *TEST 1          STF -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
(4) *
(4) * TEST, FDST MODE 0, USING STF AND AN ILLEGAL AC.
(4) *
(3) :*****
(2) 004126 000004 TST1: SCOPE
217
218 004130 AB1:
(1) 004130 104411 LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
219 004132 170001 SETF          ; SET FPS.
220 004134 012737 004174 000244 MOV #2$,FPVEC ;SET UP FOR FP TRAPS.
221 004142 012737 004150 001164 MOV #1$,STMP2
222 004150 174007 1$: STF AC0,AC7 ;INSTRUCTION SHOULD TRAP.
223
224 004152 000240 240 ; IF HERE, IT DIDN'T !.
225 004154 170200 STFPS R0
226 004156 010037 001166 MOV R0,$TMP3 ; GET FPS...
227 004162 170300 STST R0
228 004164 010037 001170 MOV R0,$TMP4 ;...AND FEC.
229 004170 104001 ERROR 1 ; ILLEGAL AC DIDN'T TRAP.
230 004172 000431 BR ABDONE
231
232 004174 022626 2$: CMP (SP)+,(SP)+ ; FIX STACK.
233 004176 012737 100000 001172 MOV #100000,$TMP5 ; SET EXP STATUS...
234 004204 012737 000002 001174 MOV #2,$TMP6 ;...AND FEC...
235 004212 012737 001172 001166 MOV #$TMP5,$TMP3 ;...AND SET POINTER.
236 004220 170200 STFPS R0
237 004222 010037 001176 MOV R0,$TMP7 ; GET FPS...
238 004226 170300 STST R0
239 004230 010037 001200 MOV R0,$TMP10 ;...AND FEC.
240 004234 012737 001176 001170 MOV #$TMP7,$TMP4 ;...AND SET POINTER.
241 004242 004537 004052 JSR R5,CHECK2
242 004246 001172 001176 $TMP5,$TMP7
243 004252 001401 BEQ ABDONE ; BR IF STATUS OK.
244
245 004254 104002 3$: ERROR 2 ; STATUS WRONG.
246
247 004256 ABDONE:
(1) 004256 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 004260 000400 BR TST2 ;:...AND PROCEED.
```

```

254 (3) *****
(4) *TEST 2 STF -- FDST MODE 1 (INDIRECT)
(4) *
(4) * TEST FDST MODE 1 (INDIRECT) USING STF INSTRUCTION.
(3) *****
(2) 004262 000004 TST2: SCOPE
255
256 004264 BB1:
(1) 004264 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.
257 004266 012700 177777 MOV #-1,R0 ;; NULL THE RECEIVING BUFFER.
258 004272 012701 004444 MOV #BBRCD,R1
259 004276 012702 000004 MOV #4,R2
260 004302 010021 1$: MOV R0,(R1)+
261 004304 077202 SOB R2,1$
262
263 004306 170011 SETD
264 004310 012700 004464 MOV #BBDAT,R0
(1) 004314 172410 LDD (R0),ACO ;; LOAD ACO
265 004316 170001 SETF
266 004320 012737 004336 001164 MOV #BB3,$TMP2
267 004326 012700 004444 MOV #BBRCD,R0 ;; FDST ADDRESS.
268 004332 010037 001166 MOV R0,$TMP3 ;; R0 BEFORE.
269 004336 174010 BB3: STF ACO,(R0) ;; TEST INSTRUCTION.
270 004340 000240 NOP
271
272 004342 020037 001166 CMP R0,$TMP3 ;; R0 CORRECT ??
273 004346 001407 BEQ 1$ ;; YES.
274 004350 013737 001164 001170 MOV $TMP2,$TMP4
275 004356 010037 001172 MOV R0,$TMP5 ;; NO, ERROR
276 004362 104004 ERROR 4
277 004364 000443 BR BBDONE
278
279 004366 012737 004464 001166 1$: MOV #BBDAT,$TMP3 ;; SAVE POINTERS.
280 004374 012737 004454 001170 MOV #BBEXP,$TMP4 ;; SAVE POINTERS.
281 004402 012737 004444 001172 MOV #BBRCD,$TMP5
282 004410 004537 004062 JSR R5,CHECK4 ;; DID BUT FD FAIL ???
283 004414 004464 004444 BBDAT,BBRCD
284 004420 001407 BEQ 2$ ;; YES.
285 004422 004537 004052 JSR R5,CHECK2 ;; NO, IS DATA CORRECT ??
286 004426 004454 004444 BBEXP,BBRCD
287 004432 001420 BEQ BBDONE ;; YES, DONE.
288
289 004434 104005 ERROR 5 ;; NO, DATA INCORRECT.
290 004436 000401 SKP1
291 004440 104006 2$: ERROR 6 ;; BUT FD FAILED.
292 004442 000414 BR BBDONE
293
294 004444 177777 177777 177777 BBRCD: .WORD -1,-1,-1,-1
295 004454 123456 023456 177777 BBEXP: .WORD 123456,23456,-1,-1
296 004464 123456 023456 034567 BBDAT: .WORD 123456,23456,34567,45671
297
298 004474 BBDONE:
(1) 004474 104412 CLRFPs ;; CLEAR FP STATUS...
(3) 004476 000400 BR TST3 ;; ...AND PROCEED.
299

```



```
306 .....  
(3) *TEST 3 STF AND STD -- FDST MODE 2 (AUTO-INCREMENT)  
(4) *  
(4) * TEST FDST MODE 2 (AUTO-INCREMENT) USING STF AND STD.  
(4) *  
(3) .....  
(2) 004500 000004 TST3: SCOPE  
307 :  
308 : FIRST THE STF.  
309 :  
310 CB1: .....  
(1) 004502 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
311 004504 012700 177777 MOV #-1,R0 ;SET UP THE OUTPUT BUFFER.  
312 004510 012701 005030 MOV #CBRCD,R1  
313 004514 012702 000004 MOV #4,R2  
314 004520 010021 1$: MOV R0,(R1)+  
315 004522 077202 SOB R2,1$  
316 :  
317 004524 112737 000106 034015 MOVB #'F,EM7X ; FIX ERROR TEXT.  
318 004532 112737 000106 034060 MOVB #'F,EM10X  
319 004540 170011 SETD  
320 004542 012700 005050 MOV #CBDAT,R0  
(1) 004546 172410 LDD (R0),ACO ;: LOAD ACO  
321 004550 170001 SETF  
322 004552 012737 004570 001164 MOV #CB3,$TMP2  
323 004560 012700 005030 MOV #CBRCD,R0 ;FDST ADDRESS.  
324 004564 010037 001166 MOV R0,$TMP3 ; R0 BEFORE.  
325 004570 174020 CB3: STF ACO,(R0)+ ;TEST INSTRUCTION.  
326 004572 000240 NOP  
327 004574 020027 005034 CMP R0,#CBRCD+4 ; R0 AUTO-INCREMENT RIGHT ??  
328 004600 001407 BEQ 1$ ; BR IF SO.  
329 004602 012737 005034 001170 MOV #CBRCD+4,$TMP4  
330 004610 010037 001172 MOV R0,$TMP5 ; R0 INCORRECT.  
331 004614 104007 ERROR 7  
332 004616 000420 BR CB20  
333 :  
334 1$: JSR R5,CHECK4 ; DATA RIGHT ??  
335 004624 005040 005030 CBXPF,CBRCD  
336 004630 001413 BEQ CB20 ; BR IF SO.  
337 004632 012737 005050 001166 MOV #CBDAT,$TMP3 ; NO, SET POINTERS.  
338 004640 012737 005040 001170 MOV #CBXPF,$TMP4  
339 004646 012737 005030 001172 MOV #CBRCD,$TMP5  
340 004654 104010 ERROR 10 ; DATA WRONG.  
341 004656 000400 BR CB20  
342 :  
343 ;NOW TEST STD MODE 2.  
344 :  
345 CB20: .....  
(1) 004660 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
346 004662 012700 005030 MOV #CBRCD,R0  
347 004666 012701 000004 MOV #4,R1  
348 004672 005020 1$: CLR (R0)+ ; CLEAR RECEIVER.  
349 004674 077102 SOB R1,1$  
350 :  
351 004676 112737 000104 034015 MOVB #'D,EM7X  
352 004704 112737 000104 034060 MOVB #'D,EM10X
```

353	004712	170011				SETD		: DOUBLE MODE.
354	004714	012700	005050			MOV	#CBDAT,RO	
(1)	004720	172410				LDD	(RO),ACO	:: LOAD ACO
355	004722	012737	004740	001164		MOV	#CB23,\$TMP2	
356	004730	012700	005030			MOV	#CBRCD,RO	: SET DESTINATION ADDRESS.
357	004734	010037	001166			MOV	RO,\$TMP3	: RO BEFORE.
358	004740	174020			CB23:	STD	ACO,(RO)+	: TEST INSTRUCTION.
359	004742	000240				NOP		
360	004744	020027	005040			CMP	RO,#CBRCD+10	: AUTO-INCR OK ??
361	004750	001407				BEQ	1\$: YES.
362	004752	012737	005040	001170		MOV	#CBRCD+10,\$TMP4	
363	004760	010037	001172			MOV	RO,\$TMP5	: RO WRONG.
364	004764	104007				ERROR	7	
365	004766	000434				BR	CBDONE	
366								
367	004770	004537	004062		1\$:	JSR	R5,CHECK4	: DATA OK ??
368	004774	005050	005030			CBDAT,CBRCD		
369	005000	001427				BEQ	CBDONE	: BR IS SO.
370	005002	012737	005050	001166		MOV	#CBDAT,\$TMP3	: NO, SET POINTERS.
371	005010	012737	005050	001170		MO,	#CBDAT,\$TMP4	
372	005016	012737	005030	001172		MOV	#CBRCD,\$TMP5	
373	005024	104010				ERROR	10	: DATA WRONG.
374	005026	000414				BR	CBDONE	
375								
376	005030	177777	177777	177777		CBRCD:	.WORD -1,-1,-1,-1	
377	005040	076543	065432	177777		CBXPF:	.WORD 76543,65432,-1,-1	
378	005050					CBXPD:		
379	005050	076543	065432	054321		CBDAT:	.WORD 76543,65432,54321,43210	
380								
381	005060					CBDONE:		
(1)	005060	104412				CLRFPS		:: CLEAR FP STATUS...
(3)	005062	000400				BR	TST4	::...AND PROCEED.
382								

```

389          ;*****
(3)          ;*TEST 4          STD -- FDST MODE 2, WITH GR7 (PC IMMEDIATE)
(4)          ;*
(4)          ;* TEST FDST MODE 2, GR7 (PC IMMEDIATE) USING STD INSTRUCTION.
(3)          ;*****
(2) 005064 000004 TST4: SCOPE
390
391 005066 DB1:
(1) 005066 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.
392 005070 012701 005164 MOV #DB2+2,R1 ;: SET DATA BLOCK FOLLOWING...
393 005074 012700 005250 MOV #DBP1,R0 ;:...TEST INSTRUCTION...
394 005100 012702 000004 MOV #4,R2
395 005104 012021 1$: MOV (R0)+,(R1)+
396 005106 077202 SOB R2,1$
397 005110 012737 005236 000004 MOV #DB04,ERRVECT ; SET BUS-ERROR TRAP.
398 005116 170011 SETD
399 005120 012700 005260 MOV #DBP2,R0
(1) 005124 172410 LDD (R0),ACO ;: LOAD ACO
400 005126 012737 005162 001164 MOV #DB2,$TMP2 ; ACO POINTER.
401 005134 012737 005260 001166 MOV #DBP2,$TMP3
402 005142 012737 005270 001170 MOV #DBEXP,$TMP4
403 005150 012737 005164 001172 MOV #DB2+2,$TMP5
404 005156 012701 005174 MOV #DB2+12,R1 ; TO CALCULATE PC AFTER.
405 005162 174027 DB2: STD ACO,(PC)+ ;:TEST INSTRUCTION.
406 005164 000241 241 ; SHOULD CHANGE TO 5741.
407 005166 005741 TST -(R1) ; 5741
408 005170 005741 TST -(R1) ; 5741
409 005172 005741 TST -(R1) ; 5741
410
411 005174 004537 004062 1$: JSR R5,CHECK4 ; CHECK DATA FIRST.
412 005200 005270 005164 DBEXP,DB2+2
413 005204 001402 BEQ 2$ ; IF OK, CHECK RETURN PC.
414 005206 104011 ERROR 11 ; DATA INCORRECT, THEREFORE...
415 005210 000433 BR DBDONE ;...PC CHECK WOULD BE INVALID.
416
417 005212 020127 005166 2$: CMP R1,#DB2+4 ; WAS PC RIGHT ??
418 005216 001406 BEQ 3$ ; YES.
419 005220 010137 001166 MOV R1,$TMP3
420 005224 012737 005166 001170 MOV #DB2+4,$TMP4
421 005232 104012 ERROR 12 ; PC WRONG AFTER STD.
422 005234 000421 3$: BR DBDONE
423
424 ; FDST FLOW FAILURE MAY RESULT IN A BUS-ERROR TRAP TO 4.
425
426 005236 011637 001164 DB04: MOV (SP),$TMP2 ; GET TRAP PC.
427 005242 022626 CMP (SP)+,(SP)+ ; FIX STACK
428 005244 104013 ERROR 13
429 005246 000414 BR DBDONE
430
431 005250 000241 005741 005741 DBP1: .WORD 241,5741,5741,5741 ; NOP AND 3 TST -(R1)'S.
432 005260 005741 000241 000241 DBP2: .WORD 5741,241,241,241 ; TST -(R1) AND 3 NOP'S.
433 005270 005741 005741 005741 DBEXP: .WORD 5741,5741,5741,5741 ; EXP FINAL DATA BLOCK.
434
435 005300 DBDONE:
(1) 005300 104412 CLRFP ;: CLEAR FP STATUS...

```

CJFPBA -- LS111/23 FPF11 DIAGNOSTIC, PART 2
CJFPBA.P11 12-FEB-81 10:27 T4

J 3
MACY11 30G(1063) 12-FEB-81 11:04 PAGE 2-5
STD -- FDST MODE 2, WITH GR7 (PC IMMEDIATE)

SEQ 0035

(3) 005302 000400

BR TST5 ::...AND PROCEED.

```
442          ;*****
(3)          ;*TEST 5          STD -- FDST MODE 4 (AUTO-DECREMENT)
(4)          ;*
(4)          ;* TEST FDST MODE 4 (AUTO-DECREMENT) USING STD INSTRUCTION.
(4)          ;*
(3)          ;*****
(2) 005304 000004 TST5: SCOPE
443
444 005306 EB1:
(1) 005306 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
445 005310 012700 177777 MOV #-1,R0
446 005314 012701 005464 MOV #EBB0,R1
447 005320 012702 000010 MOV #10,R2
448 005324 010021 1$: MOV R0,(R1)+ ; NULL THE DATA BUFFER.
449 005326 077202 SOB R2,1$
450
451 005330 012737 005452 000004 MOV #EB04,ERRVEC ; SET BUS-ERROR TRAP.
452 005336 170011 SETD
453 005340 012700 005504 MOV #EBP1,R0
(1) 005344 172410 LDD (R0),AC0 ;: LOAD AC0
454 005346 012737 005364 001164 MOV #EB2,$TMP2
455 005354 012700 005474 MOV #EBA1,R0 ;:DESTINATION ADDRESS.
456 005360 010037 001166 MOV R0,$TMP3 ;: SAVE R0 BEFORE.
457 005364 174040 EB2: STD AC0,-(R0) ;:TEST INSTRUCTION.
458 005366 000241 241
459
460 005370 020027 005464 CMP R0,#EBB0 ; AUTO-DECREMENT OK ??
461 005374 001406 BEQ 1$ ; YUP.
462 005376 012737 005464 001170 MOV #EBB0,$TMP4 ; EXPECTED R0.
463 005404 010037 001172 MOV R0,$TMP5
464 005410 104014 ERROR 14
465
466 005412 004537 004062 1$: JSR R5,CHECK4 ; DATA RIGHT ??
467 005416 005504 005464 EBP1,EBB0
468 005422 001412 BEQ 2$ ; BR IF SO.
469 005424 012737 005504 001166 MOV #EBP1,$TMP3 ; AC DATA.
470 005432 013737 001166 001170 MOV $TMP3,$TMP4 ; EXPD.
471 005440 012737 005464 001172 MOV #EBB0,$TMP5 ; RECD.
472 005446 104015 ERROR 15
473 005450 000421 2$: BR EBDONE
474
475 ;: AN FDST FAILURE MIGHT CAUSE A BUS-ERROR TRAP.
476
477 005452 011637 001164 EB04: MOV (SP),$TMP2 ; GET TRAP PC.
478 005456 022626 CMP (SP)+,(SP)+ ; FIX STACK.
479 005460 104016 ERROR 16
480 005462 000414 BR EBDONE
481
482 005464 177777 177777 177777 EBBO: .WORD -1,-1,-1,-1 ; TARGET BUFFER.
483 005474 177777 177777 177777 EBA1: .WORD -1,-1,-1,-1 ; EBA1 IS INITIAL R0.
484 005504 147250 036147 025036 EBP1: .WORD 147250,36147,25036,147250 ; AC0 DATA.
485
486 005514 EBDONE:
(1) 005514 104412 CLRFP ;: CLEAR FP STATUS...
(3) 005516 000400 BR TST6 ;: .AND PROCEED.
```

```
493 .....  
(3) *TEST 6 STD -- FDST MODE 3 (AUTO-INCR DEFERRED)  
(4) *  
(4) * TEST FDST MODE 3 (AUTO-INCREMENT DEFERRED) USING STD.  
(4) *  
(3) .....  
(2) 005520 000004 TST6: SCOPE  
494  
495 005522 FB1:  
(1) 005522 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.  
496 005524 012701 005706 MOV #FBB0,R1  
497 005530 012700 177777 MOV #-1,R0  
498 005534 012702 000012 MOV #12,R2  
499 005540 010021 1$: MOV R0,(R1)+ ; NULL THE DATA BUFFER.  
500 005542 077202 SOB R2,1$  
501  
502 005544 012737 005674 000004 MOV #FB04,ERRVEC ;SET BUS-ERROR TRAP.  
503 005552 170011 SETD  
504 005554 012700 005732 MOV #FBP1,R0  
(1) 005560 172410 LDD (R0),AC0 ;; LOAD AC0  
505 005562 013737 005606 001164 MOV FB2,$TMP2  
506 005570 012737 005706 005720 MOV #FBB0,FBA1  
507 005576 012700 005720 MOV #FBA1,R0 ;SET UP THE DESTINATION ADDRESS.  
508 005602 010037 001166 MOV R0,$TMP3 ; R0 BEFORE.  
509 005606 174030 FB2: STD AC0,@(R0)+ ;TEST INSTRUCTION.  
510 005610 000241 241  
511  
512 005612 020027 005722 CMP R0,#FBA1+2 ; AUTO-INCR CORRECT ??  
513 005616 001406 BEQ 1$ ; YES.  
514 005620 012737 005722 001170 MOV #FBA1+2,$TMP4 ; NO, EXP R0.  
515 005626 010037 001172 MOV R0,$TMP5  
516 005632 104017 ERROR 17 ; R0 BAD.  
517  
518 005634 004537 004062 1$: JSR R5,CHECK4 ; DATA CORRECT ??  
519 005640 005732 005706 FBP1,FBB0  
520 005644 001412 BEQ 2$ ; YES.  
521 005646 012737 005732 001166 MOV #FBP1,$TMP3 ; AC DATA.  
522 005654 013737 001166 001170 MOV $TMP3,$TMP4 ; EXP DATA.  
523 005662 012737 005706 001172 MOV #FBB0,$TMP5  
524 005670 104020 ERROR 20  
525 005672 000423 2$: BR FBDONE  
526  
527 ; REPORT BUS-ERROR TRAP ON TEST INSTRUCTION.  
528  
529 005674 011637 001164 FB04: MOV (SP),$TMP2 ; GET TRAP PC.  
530 005700 022626 CMP (SP)+,(SP)+  
531 005702 104021 ERROR 21  
532 005704 000416 BR FBDONE  
533  
534 005706 177777 177777 177777 FBB0: .WORD -1,-1,-1,-1  
535 005716 177777 -1  
536 005720 005706 177777 177777 FBA1: .WORD FBB0,-1,-1,-1  
537 005730 177777 -1  
538 005732 101213 141516 071727 FBP1: .WORD 101213,141516,71727,37475  
539  
540 005742 FBDONE:
```

CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
CJFPBA.P11 12-FEB-81 10:27 T6

M 3
MACY11 30G(1063) 12-FEB-81 11:04 PAGE 2-8
STD -- FDST MODE 3 (AUTO-INCR DEFERRED)

SEQ 0038

(1) 005742 104412
(3) 005744 000400

CLRFPS
BR TST7

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

```

547 .....
(3) *TEST 7          STD -- FDST MODE 5 (AUTO-DECR DEFERRED)
(4) *
(4) * TEST FDST MODE 5 (AUTO-DECREMENT DEFERRED) USING STD.
(4) *
(3) .....
(2) 005746 000004 TST7:  SCOPE
548
549 005750 GB1:
(1) 005750 104411      LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
550 005752 012701 006134  MOV      #GBB0,R1
551 005756 012700 177777  MOV      #-1,R0
552 005762 012702 000012  MOV      #12,R2
553 005766 010021 1$:      MOV      R0,(R1)+  ; NULL THE DATA BUFFER.
554 005770 077202      SOB      R2,1$
555
556 005772 012737 006122 000004  MOV      #GB04,ERRVEC ; SET BUS-ERROR TRAP.
557 006000 170011      SETD
558 006002 012700 006160      MOV      #GBP1,R0
(1) 006006 172410      LDD      (R0),AC0    ;; LOAD AC0
559 006010 013737 006034 001164  MOV      GB2,$TMP2
560 006016 012737 006134 006146  MOV      #GBB0,GBA1
561 006024 012700 006150      MOV      #GBA1+2,R0 ; SET UP THE DESTINATION ADDRESS.
562 006030 010037 001166      MOV      R0,$TMP3  ; RO BEFORE.
563 006034 174050      GB2:     STD      AC0,@-(R0) ; TEST INSTRUCTION.
564 006036 000241      241
565
566 006040 020027 006146      CMP      R0,#GBA1   ; AUTO-DECR CORRECT ??
567 006044 001406      BEQ      1$        ; YES.
568 006046 012737 006146 001170  MOV      #GBA1,$TMP4 ; NO SET EXP RO...
569 006054 010037 001172      MOV      R0,$TMP5  ; ...AND RECD RO.
570 006060 104022      ERROR    22
571
572 006062 004537 004062 1$:      JSR      R5,CHECK4  ; DATA CORRECT ??
573 006066 006160 006134      GBP1,GBB0
574 006072 001412      BEQ      2$        ; YES.
575 006074 012737 006160 001166  MOV      #GBP1,$TMP3
576 006102 013737 001166 001170  MOV      $TMP3,$TMP4
577 006110 012737 006134 001172  MOV      #GBB0,$TMP5
578 006116 104023      ERROR    23
579 006120 000423 2$:      BR      GBDONE
580
581 ; REPORT TEST INSTRUCTION TRAPPED TO 4.
582
583 006122 011637 001164 GB04:   MOV      (SP),$TMP2
584 006126 022626      CMP      (SP)+,(SP)+
585 006130 104024      ERROR    24
586 006132 000416      BR      GBDONE
587
588 006134 177777 177777 177777 GBB0:   .WORD   -1,-1,-1,-1
589 006144 177777      -1
590 006146 006134 177777 177777 GBA1:   .WORD   GBB0,-1,-1,-1
591 006156 177777      -1
592 006160 020212 023242 026273 GBP1:   .WORD   20212,23242,26273,31323
593
594 006170      GBDONE:

```


CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
CJFPBA.P11 12-FEB-81 10:27 T7

MACY11 30G(1063) ^{B 4} 12-FEB-81 11:04 PAGE 2-10
STD -- FDST MODE 5 (AUTO-DECR DEFERRED)

SEQ 0040

(1) 006170 104412
(3) 006172 000400

CLRFPS
BR TST10

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

```
601 .....  
(3) : *TEST 10 STD -- FDST MODE 6 (INDEX)  
(4) : *  
(4) : * TEST FDST MODE 6 (INDEX) USING STD.  
(4) : *  
(3) : .....  
(2) 006174 000004 TST10: SCOPE  
602  
603 006176 HB1: LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
(1) 006176 104411 MOV #HBB0,R1  
604 006200 012701 006356 MOV #-1,R0  
605 006204 012700 177777 MOV #4,R2  
606 006210 012702 000004 1$: MOV R0,(R1)+ ; NULL THE DATA BUFFER.  
607 006214 010021 SOB R2,1$  
608 006216 077202  
609  
610 006220 012737 006344 000004 MOV #HB04,ERRVEC ;SET BUS-ERROR TRAP.  
611 006226 170011 SETD  
612 006230 012700 006370 MOV #HBP1,R0  
(1) 006234 172410 LDD (R0),AC0 ;: LOAD AC0  
613 006236 012737 006254 001164 MOV #HB2,$TMP2  
614 006244 012700 006115 MOV #HBB0-241,R0 ;SET UP THE DESTINATION ADDRESS.  
615 006250 010037 001166 MOV R0,$TMP3  
616 006254 174060 000241 HB2: STD AC0,241(R0) ;TEST INSTRUCTION.  
617 006260 000241 241  
618  
619 006262 020037 001166 CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.  
620 006266 001406 BEQ 1$ ; IT IS.  
621 006270 013737 001166 001170 MOV $TMP3,$TMP4 ; IT ISN'T.  
622 006276 010037 001172 MOV R0,$TMP5  
623 006302 104025 ERROR 25  
624  
625 006304 004537 004062 1$: JSR R5,CHECK4 ; DATA CORRECT ??  
626 006310 006370 006356 HBP1,HBB0  
627 006314 001412 BEQ 2$ ; YES.  
628 006316 012737 006370 001166 MOV #HBP1,$TMP3 ; NO.  
629 006324 013737 001166 001170 MOV $TMP3,$TMP4  
630 006332 012737 006356 001172 MOV #HBB0,$TMP5  
631 006340 104026 ERROR 26  
632 006342 000416 2$: BR HBDONE  
633  
634 : REPORT TEST INSTRUCTION TRAPPED.  
635 :  
636 006344 011637 001164 HB04: MOV (SP),$TMP2  
637 006350 022626 CMP (SP)+,(SP)+  
638 006352 104027 ERROR 27  
639 006354 000411 BR HBDONE  
640  
641 006356 177777 177777 177777 HB00: .WORD -1,-1,-1,-1  
642 006366 177777 -1  
643 006370 030313 023334 025262 HB01: .WORD 30313,23334,25262,74041  
644  
645 006400 HBDONE:  
(1) 006400 104412 CLRFPs ;: CLEAR FP STATUS...  
(3) 006402 000400 BR TST11 ;:...AND PROCEED.
```

```
652 .....  
(3) : *TEST 11 STD -- FDST MODE 7 (INDEX DEFERRED)  
(4) : *  
(4) : * TEST FDST MODE 7 (INDEX DEFERRED) USING STD.  
(4) : *  
(3) : .....  
(2) 006404 000004 TST11: SCOPE  
653  
654 006406 JB1: LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
(1) 006406 104411 MOV #JBB0,R1  
655 006410 012701 006574 MOV #-1,R0  
656 006414 012700 177777 MOV #4,R2  
657 006420 012702 000004 1$: MOV R0,(R1)+ ; NULL THE DATA BUFFER.  
658 006424 010021 SOB R2,1$  
659 006426 077202  
660  
661 006430 012737 006562 000004 MOV #JB04,ERRVEC ; SET BUS-ERROR TRAP.  
662 006436 170011 SETD  
663 006440 012700 006606 MOV #JBP1,R0  
(1) 006444 172410 LDD (R0),AC0 ;: LOAD AC0  
664 006446 012737 006472 001164 MOV #JB2,$TMP2  
665 006454 012737 006574 006620 MOV #JBB0,JBA1  
666 006462 012700 006357 MOV #JBA1-241,R0 ;SET UP THE DESTINATION ADDRESS.  
667 006466 010037 001166 MOV R0,$TMP3  
668 006472 174070 000241 JB2: STD AC0,@241(R0) ;TEST INSTRUCTION.  
669 006476 000241 241  
670  
671 006500 020037 001166 CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.  
672 006504 001406 BEQ 1$ ; BR IF SO.  
673 006506 013737 001166 001170 MOV $TMP3,$TMP4 ; IT ISN'T.  
674 006514 010037 001172 MOV R0,$TMP5  
675 006520 104030 ERROR 30  
676  
677 006522 004537 004062 1$: JSR R5,CHECK4 ; DATA CORRECT ??  
678 006526 006606 006574 JBP1,JBB0  
679 006532 001412 BEQ 2$ ; YES.  
680 006534 012737 006606 001166 MOV #JBP1,$TMP3  
681 006542 013737 001166 001170 MOV $TMP3,$TMP4  
682 006550 012737 006574 001172 MOV #JBB0,$TMP5  
683 006556 104031 ERROR 31  
684 006560 000420 2$: BR JBDONE  
685  
686 : REPORT TEST INSTRUCTION TRAPPED TO 4.  
687 :  
688 006562 011637 001164 JB04: MOV (SP),$TMP2  
689 006566 022626 CMP (SP)+,(SP)+  
690 006570 104032 ERROR 32  
691 006572 000413 BR JBDONE  
692  
693 006574 177777 177777 177777 JBB0: .WORD -1,-1,-1,-1  
694 006604 177777 -1  
695 006606 041424 034445 046475 JBP1: .WORD 41424,34445,46475,51525  
696 006616 177777 -1  
697 006620 006574 JBA1: .WORD JBB0  
698  
699 006622 JBDONE:
```

CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
CJFPBA.P11 12-FEB-81 10:27 T11

MACY11 30G(1063) ^{E 4} 12-FEB-81 11:04 PAGE 2-13
STD -- FDST MODE 7 (INDEX DEFERRED)

EQ 0043

(1) 006622 104412
(3) 006624 000400

CLRFPS
BR TST12

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

```
706 .....  
(3) *TEST 12 STCFD AND STCDF -- FDST MODE 1  
(4) *  
(4) * TEST THE STORE AND CONVERT (STCFD AND STCDF) INSTRUCTIONS.  
(4) *  
(3) .....  
(2) 006626 000004 TST12: SCOPE  
707 .....  
708 : NOTE: THIS TEST IS A COMBINATION OF TESTS 12 (STCFD)  
709 : AND 13 (STCDF) AS FOUND IN FP11-A, PART 3.  
710 :  
711 : FIRST DO STCFD (SINGLE FLOAT TO DOUBLE FLOAT).  
712 :  
713 006630 012737 042106 035360 KB1: MOV #'FD,EM34X ; SET ERROR TEXT FOR STCFD.  
714 006636 004737 007304 JSR PC,KBSUB  
715 006642 000000 000000 000000 .WORD 0,0,0,0 ; AC = 0.  
716 006652 000000 000000 000000 .WORD 0,0,0,0 ; EXP RESULT.  
717 006662 047017 .WORD 47017 ; FPS BEFORE STCFD.  
718 006664 047004 000000 .WORD 47004,0 ; EXP FPS AND FEC AFTER.  
719 :  
720 :  
721 006670 004737 007304 KB2: JSR PC,KBSUB  
722 006674 017203 142536 177777 .WORD 17203,142536,-1,-1 ; AC  
723 006704 017203 142536 000000 .WORD 17203,142536,0,0 ; EXP RESULT.  
724 006714 040017 .WORD 40017 ; FPS BEFORE.  
725 006716 040000 000000 .WORD 40000,0 ; EXP FPS,FEC AFTER.  
726 :  
727 :  
728 006722 004737 007304 KB3: JSR PC,KBSUB  
729 006726 050717 027374 177777 .WORD 50717,27374,-1,-1 ; AC.  
730 006736 050717 027374 000000 .WORD 50717,27374,0,0 ; EXP RESULT.  
731 006746 047017 .WORD 47017 ; FPS BEFORE.  
732 006750 047000 000000 .WORD 47000,0 ; FPS, FEC AFTER.  
733 :  
734 :  
735 006754 004737 007304 KB4: JSR PC,KBSUB  
736 006760 020212 032425 177777 .WORD 20212,32425,-1,-1 ; AC  
737 006770 020212 032425 000000 .WORD 20212,32425,0,0 ; EXP RESULT.  
738 007000 040017 .WORD 40017 ; FPS BEFORE.  
739 007002 040000 000000 .WORD 40000,0 ; EXP FPS,FEC.  
740 :  
741 :  
742 007006 004737 007304 KB5: JSR PC,KBSUB  
743 007012 121314 151617 177777 .WORD 121314,151617,-1,-1 ; AC  
744 007022 121314 151617 000000 .WORD 121314,151617,0,0 ; EXP RES.  
745 007032 040017 .WORD 40017 ; FPS BEFORE.  
746 007034 040010 000000 .WORD 40010,0 ; FPS,FEC AFTER.  
747 :  
748 : NOW CHANGE OVER TO STCDF (DOUBLE TO SINGLE FLOAT).  
749 :  
750 007040 012737 043104 035360 KB6: MOV #'DF,EM34X ; CHANGE ERROR TEXT.  
751 007046 004737 007304 JSR PC,KBSUB  
752 007052 000000 000000 000000 .WORD 0,0,0,0 ; AC = 0.  
753 007062 000000 000000 177777 .WORD 0,0,-1,-1 ; EXP RESULT.  
754 007072 047217 .WORD 47217 ; FPS BEFORE.  
755 007074 047204 000000 .WORD 47204,0 ; FPS, FEC AFTER.
```

```
756  
757  
758  
759 007100 004737 007304  
760 007104 067574 073727 170777  
761 007114 067574 073730 177777  
762 007124 040217  
763 007126 040200 000000  
764  
765  
766  
767 007132 004737 007304  
768 007136 067574 073727 170777  
769 007146 067574 073727 177777  
770 007156 040257  
771 007160 040240 000000  
772  
773  
774  
775 007164 004737 007304  
776 007170 077777 177777 100000  
777 007200 000000 000000 177777  
778 007210 047217  
779 007212 147206 000010  
780  
781 007216 004737 007304  
782 007222 077777 177777 100000  
783 007232 000000 000000 177777  
784 007242 046217  
785 007244 046206 000000  
786  
787  
788  
789 007250 004737 007304  
790 007254 177777 177777 100000  
791 007264 100000 000000 177777  
792 007274 047217  
793 007276 147216 000010  
794  
795 007302 000512  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811
```

CONVERT AND ROUND.

KB7: JSR PC,KBSUB
.WORD 67574,73727,170777,67574 ; AC
.WORD 67574,73730,-1,-1 ; EXP RESULT.
.WORD 40217 ; FPS (FT = 0).
.WORD 40200,0 ; FPS, FEC AFTER.

CONVERT AND TRUNCATE.

KB8: JSR PC,KBSUB
.WORD 67574,73727,170777,67574 ; AC
.WORD 67574,73727,-1,-1 ; EXP RESULT.
.WORD 40257 ; FPS (FT = 1).
.WORD 40240,0 ; FPS, FEC AFTER.

AC TOO LARGE FOR D TO F CONVERSION.

KB9: JSR PC,KBSUB
.WORD 77777,-1,100000,0 ; AC
.WORD 0,0,-1,-1 ; EXP RESULT.
.WORD 47217 ; FPS, FIV = 1.
.WORD 147206,10 ; FPS, FEC AFTER.

KB10: JSR PC,KBSUB
.WORD 77777,-1,100000,0 ; SAME OPERANDS, FIV = 0.
.WORD 0,0,-1,-1 ; AC
.WORD 46217 ; EXP RESULT.
.WORD 46206,0 ; FPS, FIV = 0.
 ; FPS, FEC AFTER.

AC TOO LARGE AND NEGATIVE. (FIUV = 1).

KB10: JSR PC,KBSUB
.WORD -1,-1,100000,0 ; AC
.WORD 100000,0,-1,-1 ; EXP RESULT.
.WORD 47217 ; FPS (FIUV = 1).
.WORD 147216,10 ; FPS, FEC AFTER.

BR KBDONE

SUBROUTINE TO SET-UP, EXECUTE, AND CHECK RESULTS OF
STCFD AND/OR STCDF INSTRUCTIONS.
NOTE THAT THE CONVERSION (F => D, OR D => F) IS DEPENDANT
ON THE PREVAILING FP MODE, AND NOT ON THE OPCODE ITSELF.
I.E. OPCODE STCFD = STCDF = 176000

CALL:

JSR PC,KBSUB
.WORD X,X,X,X ; AC OPERAND
.WORD X,X,X,X ; EXPECTED RESULT
.WORD X ; FPS BEFORE EXECUTION
.WORD X,X ; EXP FPS, FEC AFTER.
 ; RETURN TO CALL+26

NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THAT OF THE

```

812 ; CALLING SEQUENCE AND NOT THAT OF THE CONVERT INSTRUCTION.
813 ;
814 007304 012637 001164 KBSUB: MOV (SP)+,$TMP2 ; SAVE CALL PC AS ERROR PC.
815 007310 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
816 007312 012700 177777 MOV #-1,R0
817 007316 012701 007514 MOV #20$,R1
818 007322 012702 000004 MOV #4,R2
819 007326 010021 1$: MOV R0,(R1)+ ; NULL RECEIVING BUFFER.
820 007330 077202 SOB R2,1$
821
822 007332 013701 001164 MOV $TMP2,R1 ; SET ARG POINTER.
823 007336 170011 SETD ;
824 007340 010100 MOV R1,R0 ;LOAD ACO.
825 007342 172410 LDD (R0),ACO
826 007344 010037 001166 MOV R0,$TMP3 ; SAVE AC...
827 007350 062700 000010 ADD #10,R0
828 007354 010037 001170 MOV R0,$TMP4 ;...EXP RESULT...
829 007360 012737 007514 001172 MOV #20$, $TMP5 ;...RECD RESULT...
830 007366 062700 000012 ADD #12,R0
831 007372 010037 001174 MOV R0,$TMP6 ;...EXP STATUS...
832 007376 012737 007524 001176 MOV #21$, $TMP7 ;...AND RECD STATUS POINTERS.
833 007404 016100 000020 MOV 20(R1),R0 ; INITIALIZE THE FPS.
834 007410 170100 LDFPS R0
835 007412 012700 007514 MOV #20$,R0 ; SET FDST.
836 007416 176010 2$: STCFD ACO,(R0) ; CONVERT F -> D OR D -> F.
837 007420 000241 241
838
839 007422 170200 STFPS R0
840 007424 010037 007524 MOV R0,21$ ; GET FPS...
841 007430 005000 CLR R0
842 007432 005761 000024 TST 24(R1) ;...IF EXP'D FEC IS NON-ZERO...
843 007436 001401 BEQ .+4
844 007440 170300 STST R0 ;...GET FEC TOO.
845 007442 010037 007526 MOV R0,21$+2
846
847 007446 013737 001170 007460 3$: MOV $TMP4,4$ ; SET EXP DATA ADDRESS...
848 007454 004537 004062 JSR R5,CHECK4 ;...AND CHECK RESULTING DATA.
849 007460 000000 007514 4$: 0,20$
850 007464 001010 BNE 6$
851 007466 013737 001174 007500 MOV $TMP6,5$ ; SET EXP STATUS ADDRESS...
852 007474 004537 004052 JSR R5,CHECK2 ;...AND CHECK RESULTING STATUS.
853 007500 000000 007524 5$: 0,21$
854 007504 001401 BEQ 7$
855
856 007506 104034 6$: ERROR 34 ; STATUS OR RESULT WRONG.
857
858 007510 000161 000026 7$: JMP 26(R1) ; RETURN.
859
860 007514 177777 177777 177777 20$: .WORD -1,-1,-1,-1 ; RECEIVE CONVERTED DATA.
861 007524 177777 177777 21$: .WORD -1,-1 ; RECEIVED FPS AND FEC.
862
863 007530 KBDONE:
(1) 007530 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 007532 000400 BR TST13 ;:...AND PROCEED.

```

```

870 .....
(3) *TEST 13          STCFD -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
(4) *
(4) * TEST STCFD WITH AN ILLEGAL ACCUMULATOR (AC6).
(3) .....
(2) 007534 000004 TST13: SCOPE
871
872 007536 LB1:
(1) 007536 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
873 007540 012700 040000 MOV #40000,R0 ; F MODE, INTERRUPTS DISABLED.
874 007544 170100 LDFPS R0
875 007546 012737 007554 001164 MOV #LB2,$TMP2
876 007554 176006 LB2: STCFD AC0,AC6 ; SHOULD CAUSE AN ERROR.
877 007556 000241 241
878
879 007560 170200 STFPS R0
880 007562 010037 001172 MOV R0,$TMP5 ; GET FPS...
881 007566 170300 STST R0
882 007570 010037 001174 MOV R0,$TMP6 ; ...AND FEC.
883 007574 004537 004052 JSR R5,CHECK2 ; CHECK STATUS.
884 007600 007626 001172 10$,$TMP5
885 007604 001407 BEQ 2$
886
887 007606 012737 007626 001166 1$: MOV #10$,$TMP3
888 007614 012737 001172 001170 MOV #$TMP5,$TMP4
889 007622 104035 ERROR 35
890 007624 000402 2$: BR LBDONE
891
892 007626 140000 000002 10$: .WORD 140000,2 ; EXP FPS, FEC.
893
894 007632 LBDONE:
(1) 007632 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 007634 000400 BR TSI14 ;: ...AND PROCEED.

```



```

901 .....
(3) *TEST 14 CLRF AND CLRD -- FDST MODE 1
(4) *
(4) *TEST THE CLRF AND CLRD INSTRUCTIONS.
(4) *
(3) .....
(2) 007636 000004 TST14: SCOPE
902
903 007640 112737 000106 035527 MB1: MOVB #'F,EM36X ; DO F MODE FIRST.
904 007646 012705 000017 MOV #17,R5 ; INITIAL FPS.
905 007652 012737 000004 001170 MOV #4,$TMP4 ; EXP FPS.
906 007660 012737 010046 001172 MOV #MBEF,$TMP5 ; EXP DATA.
907 007666 000413 BR MB3
908 007670 112737 000104 035527 MB2: MOVB #'D,EM36X ; NOW DO D MODE.
909 007676 012705 000217 MOV #217,R5
910 007702 012737 000204 001170 MOV #204,$TMP4
911 007710 012737 010056 001172 MOV #MBED,$TMP5
912
913 MB3:
(1) 007716 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
914 007720 170105 LDFPS R5 ; INITIALIZE FPS.
915 007722 012701 010036 MOV #MBDB,R1
916 007726 012702 000004 MOV #4,R2
917 007732 012721 177777 1$: MOV #-1,(R1)+ ; SET DATA NON-ZERO.
918 007736 077203 SOB R2,1$
919 007740 012700 010036 MOV #MBDB,R0 ; SET FDST.
920 007744 170410 2$: CLRF (R0) ; CLRF OR CLRD.
921 007746 000241 241
922 007750 170205 STFPS R5
923 007752 010537 001166 MOV R5,$TMP3 ; GET FPS.
924 007756 023737 001166 001170 CMP $TMP3,$TMP4 ; FPS RIGHT ??
925 007764 001010 BNE 4$ ; NO.
926 007766 013737 001172 010000 MOV $TMP5,3$
927 007774 004537 004062 JSR R5,CHECK4 ; YES, CLEAR OK ??
928 010000 000000 010036 3$: O,MBDB
929 010004 001407 BEQ 5$ ; BR IF BOTH OK.
930
931 010006 012737 007744 001164 4$: MOV #2,$TMP2
932 010014 012737 010036 001174 MOV #MBDB,$TMP6
933 010022 104036 ERROR 36
934
935 010024 023727 001170 000204 5$: CMP $TMP4,#204 ; D MODE DONE ??
936 010032 001316 BNE MB2 ; NO, DO IT NOW.
937 010034 000414 BR MBDONE ; YES, QUIT.
938
939 010036 177777 177777 177777 MBDB: .WORD -1,-1,-1,-1 ; DATA BUFFER TO CLEAR.
940 010046 000000 000000 177777 MBEF: .WORD 0,0,-1,-1 ; EXP'D F MODE CLEAR.
941 010056 000000 000000 000000 MBED: .WORD 0,0,0,0 ; EXP'D D MODE CLEAR.
942
943 MBDONE:
(1) 010066 104412 CLRFPS ;; CLEAR FP STATUS...
(3) 010070 000400 BR TST15 ;;...AND PROCEED.
  
```

```

950
(3)
(4)
(4)
(4)
(3)
(2) 010072 000004
951
952 010074
(1) 010074 104411
953 010076 012700 040200
954 010102 170100
955 010104 012737 010112 001164
956 010112 170407
957 010114 000241
958 010116 012737 140200 001172
959 010124 012737 000002 001174
960 010132 170200
961 010134 010037 001176
962 010140 170300
963 010142 010037 001200
964 010146 004537 004052
965 010152 001172 001176
966 010156 001407
967
968 010160 012737 001172 001166
969 010166 012737 001176 001170
970 010174 104037
971
972 010176
(1) 010176 104412
(3) 010200 000400
  
```

```

*****
*TEST 15      CLRD -- FDST MODE 0, WITH ILLEGAL ACCUMULATOR
*
* TEST CLRD WITH AN ILLEGAL ACCUMULATOR (AC7).
*
*****
TST15: SCOPE

NB1:
LUPERR                                ;; LOOP HERE ON ERROR IF SWR9 - 1.
MOV #40200,R0                          ; DOUBLE, INTERRUPT DISABLED.
LDFPS R0
MOV #1$, $TMP2
1$: CLRD AC7                            ; TEST INSTRUCTION.
241
MOV #140200, $TMP5                      ; SET EXP STATUS
MOV #2, $TMP6
STFPS R0
MOV R0, $TMP7                          ; GET FPS...
STST R0
MOV R0, $TMP10                          ; ...AND FEC.
JSR R5, CHECK2                          ; STATUS CORRECT ??
$TMP5, $TMP7
BEQ NBDONE ; EXIT IF SO.

MOV # $TMP5, $TMP3
MOV # $TMP7, $TMP4
ERROR 37

NBDONE:
CLR FPS
BR TST16                                ;; CLEAR FP STATUS...
;; ...AND PROCEED.
  
```

980
(3)
(4)
(4)
(4)
(4)
(3)
(2) 010202 000004
981
982 010204
(1) 010204 104411
983 010206 012700 040200
984 010212 170100
985 010214 012737 010222 001164
986 010222 170707
987 010224 000241
988 010226 012737 140200 001172
989 010234 012737 000002 001174
990 010242 170200
991 010244 010037 001176
992 010250 170300
993 010252 010037 001200
994 010256 004537 004052
995 010262 001172 001176
996 010266 001407
997
998 010270 012737 001172 001166
999 010276 012737 001176 001170
1000 010304 104040
1001
1002 010306
(1) 010306 104412
(3) 010310 000400

```
*****  
*TEST 16 SPECIAL FDST FLOW, USING NEGD MODE 0 WITH ILLEGAL AC7  
*  
* TEST THAT THE SPECIAL FDST FLOW USED BY ABS, NEG, AND TST  
* WILL TRAP IF AN ILLEGAL MODE 0 AC IS USED.  
*  
*****  
TST16: SCOPE  
  
PB1:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
MOV #40200,R0  
LDFPS R0 ; SET FPS, DOUBLE, FID = 1.  
MOV #1$, $TMP2  
1$: NEGD AC7 ;TEST INSTRUCTION.  
241  
MOV #140200,$TMP5 ; EXP FPS...  
MOV #2,$TMP6 ;...AND FEC.  
STFPS R0 ;GET FPS...  
MOV R0,$TMP7 ;...AND FEC.  
STST R0  
MOV R0,$TMP10  
JSR R5,CHECK2  
$TMP5,$TMP7  
BEQ PBDONE ; BR IF STATUS IS RIGHT.  
  
MOV #$TMP5,$TMP3  
MOV #$TMP7,$TMP4  
ERROR 40  
  
PBDONE:  
CLRFPS ;: CLEAR FP STATUS...  
BR TST17 ;...AND PROCEED.
```

```
1012 .....  
(3) *TEST 17 SPECIAL FDST FLOW, USING NEGD MODE 0  
(4) *  
(4) * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST  
(4) * INSTRUCTIONS, USING NEGD, MODE 0  
(4) * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) - 0,  
(4) * AND AGAIN WITH E(FDST) NON-ZERO.  
(3) .....  
(2) 010312 000004 TST17: SCOPE  
1013 .....  
1014 * THIS TEST COMBINES TESTS 20 AND 32 FROM FP11-A, PART 3.  
1015 .....  
1016 010314 012737 010474 001172 QB1: MOV #QB10,$TMP5 ; INITIAL AC DATA.  
1017 010322 012737 010504 001174 MOV #QB11,$TMP6 ; EXPD RESULT.  
1018 010330 012737 000204 001170 MOV #204,$TMP4 ; EXP FPS.  
1019 010336 000411 BR QB3  
1020 010340 012737 010514 001172 QB2: MOV #QB12,$TMP5 ; INITIAL AC (EXPN NON-ZERO).  
1021 010346 012737 010524 001174 MOV #QB13,$TMP6  
1022 010354 012737 000210 001170 MOV #210,$TMP4  
1023 .....  
1024 010362 QB3: LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
(1) 010362 104411 SETD  
1025 010364 170011 MOV $TMP5,R0 ; LOAD UP INITIAL AC.  
1026 010366 013700 001172 LDD (R0),AC0  
1027 010372 172410 MOV #1$, $TMP2 ;: TEST INSTRUCTION.  
1028 010374 012737 010402 001164 1$: NEGD AC0 ; GET FPS.  
1029 010402 170700 STFPS R5 ; INSURANCE.  
1030 010404 000241 MOV R5,$TMP3 ;: STORE AC0  
1031 010406 170205 SETD ; SET POINTER FOR CHECKER.  
1032 010410 010537 001166 MOV #QB14,R0 ; CHECK RESULT.  
1033 010414 170011 STD AC0,(R0) ; BR IF WRONG.  
1034 010416 012700 010534 MOV R0,$TMP7 ; CHECK FPS.  
(1) 010422 174010 MOV $TMP6,2$ ; BR IF BOTH OK.  
1035 010424 010037 001176 JSR R5,CHECK4  
1036 010430 013737 001174 010442 2$: 0,QB14  
1037 010436 004537 004062 BNE 3$  
1038 010442 000000 010534 CMP $TMP3,$TMP4 ; 2ND PASS DONE ??  
1039 010446 001004 BNE QB2 ; NO, DO IT NOW.  
1040 010450 023737 001166 001170 BEQ 4$  
1041 010456 001401 3$: ERROR 41  
1042 .....  
1043 010460 104041 4$: CMP $TMP4,#210 ;  
1044 ..... BNE QB2 ;  
1045 010462 023727 001170 000210 BR QBDONE ;  
1046 010470 001323  
1047 010472 000424  
1048 .....  
1049 010474 000000 131415 161710 QB10: .WORD 0,131415,161710,111213 ; EXPONENT = 0.  
1050 010504 000000 000000 000000 QB11: .WORD 0,0,0,0 ; EXP'D RESULT.  
1051 .....  
1052 010514 013572 046013 057246 QB12: .WORD 013572,46013,57246,13570 ; POS NON-ZERO.  
1053 010524 113572 046013 057246 QB13: .WORD 113572,46013,57246,13570 ; EXP'D RESULT.  
1054 .....  
1055 010534 000000 000000 000000 QB14: .WORD 0,0,0,0 ; RECEIVED DATA UNDER TEST.  
1056 .....
```

CJFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
CJFPBA.P11 12-FEB-81 10:27 T17

MACY11 30G(1063) ^{N 4} 12-FEB-81 11:04 PAGE 2-22
SPECIAL FDST FLOW, USING NEG D MODE 0

SEG 0052

1057 010544
(1) 010544 104412
(3) 010546 000400

QBDONE:

CLRFPS
BR TST20

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

1

```
1059 (3) :*****
      (4) :*TEST 20 SPECIAL FDST FLOW, USING NEGD MODE 1
      (4) :*
      (4) :* TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
      (4) :* INSTRUCTIONS, USING NEGD, MODE 1.
      (4) :* EXECUTE THE TEST TWICE, FIRST WITH E(FDST) - 0,
      (4) :* AND AGAIN WITH E(FDST) NON-ZERO.
      (3) :*****
      (2) 010550 000004 TST20: SCOPE
1060 :
1061 : THIS COMBINES TESTS 21 AND 33 FROM FP11-A, PART 3.
1062 :
1063 010552 012737 011012 001172 RB1: MOV #RB10,$TMP5 ; INITIAL DATA (EXPONENT 0).
1064 010560 012737 011022 001174 MOV #RB11,$TMP6 ; EXPECTED RESULT.
1065 010566 012737 000204 001170 MOV #204,$TMP4 ; EXPECTED FPS.
1066 010574 000411 BR RB3
1067 010576 012737 011032 001172 RB2: MOV #RB12,$TMP5 ; INITIAL DATA (POS NON-ZERO).
1068 010604 012737 011042 001174 MOV #RB13,$TMP6
1069 010612 012737 000210 001170 MOV #210,$TMP4
1070 :
1071 010620 RB3:
      (1) 010620 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
1072 010622 170011 SETD
1073 010624 013700 001172 MOV $TMP5,R0
1074 010630 012701 011052 MOV #RB14,R1
1075 010634 012702 000004 MOV #4,R2
1076 010640 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
1077 010642 077202 SOB R2,1$
1078 :
1079 010644 012737 011000 000004 MOV #RB04,ERRVEC ; IN CASE IT TRAPS.
1080 010652 012737 010670 001164 MOV #2$,$TMP2
1081 010660 012700 011052 MOV #RB14,R0 ; SET FDST.
1082 010664 010037 001166 MOV R0,$TMP3 ; SAVE R0 BEFORE.
1083 010670 170710 2$: NEGD (R0) ; TEST INSTRUCTION.
1084 010672 000241 241
1085 010674 020037 001166 CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.
1086 010700 001407 BEQ 3$ ; BR IF SO.
1087 010702 013737 001166 001170 MOV $TMP3,$TMP4
1088 010710 010037 001172 MOV R0,$TMP5
1089 010714 104042 ERROR 42 ; R0 WRONG.
1090 010716 000461 BR RBDONE ; AND QUIT.
1091 :
1092 010720 170200 3$: STFPS R0 ; GET FPS.
1093 010722 010037 001166 MOV R0,$TMP3
1094 010726 013737 001174 010740 MOV $TMP6,4$ ; SET EXP'D POINTER.
1095 010734 004537 004062 JSR R5,CHECK4 ; DATA CORRECT ??
1096 010740 000000 011052 4$: O,RB14
1097 010744 001004 BNE 5$ ; NO.
1098 010746 023737 001166 001170 CMP $TMP3,$TMP4 ; FPS CORRECT ??
1099 010754 001404 BEQ 6$ ; BR IF BOTH OK.
1100 010756 012737 011052 001176 5$: MOV #RB14,$TMP7
1101 010764 104043 ERROR 43 ; FPS OR RESULT WRONG.
1102 :
1103 010766 023727 001170 000210 6$: CMP $TMP4,#210 ; 2ND PASS DONE ??
1104 010774 001300 BNE RB2 ; NO, DO IT NOW.
```

```
1105 010776 000431 BR RBDONE ; YES, EXIT.
1106
1107 ; WE'RE HERE IF IT TRAPPED.
1108
1109 011000 011637 001164 RB04: MOV (SP), $TMP2 ; GET TRAP PC.
1110 011004 022626 CMP (SP)+, (SP)+
1111 011006 104044 ERROR 44
1112 011010 000424 BR RBDONE
1113
1114 011012 000177 167574 137271 RB10: .WORD 177,167574,137271,107675 ; EXPONENT = 0.
1115 011022 000000 000000 000000 RB11: .WORD 0,0,0,0 ; EXPECTED RESULT.
1116
1117 011032 023245 026720 122324 RB12: .WORD 023245,26720,122324,52672 ; POS NON-ZERO.
1118 011042 123245 026720 122324 RB13: .WORD 123245,26720,122324,52672 ; EXPECTED RESULT.
1119
1120 011052 000000 000000 000000 RB14: .WORD 0,0,0,0 ; DATA BUFFER.
1121
1122 011062 RBDONE:
(1) 011062 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 011064 000400 BR TST21 ;:...AND PROCEED.
```

```

1124 .....
(3) : *TEST 21 SPECIAL FDST FLOW, USING ABSD MODE 2
(4) : *
(4) : * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
(4) : * INSTRUCTIONS, USING ABSD, MODE 2.
(4) : * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
(4) : * AND AGAIN WITH E(FDST) NON-ZERO.
(4) : *
(3) : .....
(2) 011066 000004 TST21: SCOPE
1125 :
1126 : THIS COMBINES TESTS 22 AND 34 FROM FP11-A, PART 3.
1127 :
1128 011070 012737 011330 001172 SB1: MOV #SB10,$TMP5 ; INITIAL DATA (EXP 0).
1129 011076 012737 011340 001174 MOV #SB11,$TMP6 ; EXPECTED RESULT.
1130 011104 012737 000204 001170 MOV #204,$TMP4 ; EXPECTED FPS.
1131 011112 000411 BR SB3
1132 011114 012737 011350 001172 SB2: MOV #SB12,$TMP5 ; INITIAL DATA (EXP NON-ZERO).
1133 011122 012737 011360 001174 MOV #SB13,$TMP6 ; EXPECTED.
1134 011130 012737 000200 001170 MOV #200,$TMP4 ; EXPECTED FPS.
1135 :
1136 011136 SB3:
(1) 011136 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
1137 011140 170011 SETD
1138 011142 013700 001172 MOV $TMP5,R0
1139 011146 012701 011400 MOV #SB14,R1
1140 011152 012702 000004 MOV #4,R2
1141 011156 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
1142 011160 077202 SOB R2,1$
1143 :
1144 011162 012737 011316 000004 MOV #SB04,ERRVEC ; SET BUS-ERR TRAP.
1145 011170 012737 011206 001164 MOV #2$, $TMP2
1146 011176 012700 011400 MOV #SB14,R0 ; SET UP THE OPERAND ADDRESS.
1147 011202 010037 001166 MOV R0,$TMP3 ; SAVE R0 BEFORE.
1148 011206 170620 2$: ABSD (R0)+ ; TEST INSTRUCTION.
1149 011210 000241 241
1150 011212 020027 011410 CMP R0,#SB14+10 ; AUTO-INCR OK ??
1151 011216 001407 BEQ 3$ ; BR IF SO.
1152 011220 012737 011410 001170 MOV #SB14+10,$TMP4
1153 011226 010037 001172 MOV R0,$TMP5
1154 011232 104045 ERROR 45 ; R0 INCORRECT.
1155 011234 000465 BR SBDONE
1156 :
1157 011236 170200 3$: STFPS R0 ; GET FPS.
1158 011240 010037 001166 MOV R0,$TMP3
1159 011244 013737 001174 011256 MOV $TMP6,4$ ; SET EXP'D POINTER.
1160 011252 004537 004062 4$: ISR R5,CHECK4 ; AND CHECK FINAL DATA.
1161 011256 000000 011400 4$: U,SB14
1162 011262 001004 BNE 5$ ; BR IF WRONG.
1163 011264 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK FPS.
1164 011272 001404 BEQ 6$ ; BR IF BOTH OK.
1165 011274 012737 011400 001176 5$: MOV #SB14,$TMP7
1166 011302 104046 ERROR 46 ; DATA OR STATUS WRONG.
1167 :
1168 011304 023727 001170 000200 6$: CMP $TMP4,#200 ; 2ND PASS ???
1169 011312 001300 BNE SB2 ; NO, DO IT NOW.

```



```
1170 011314 000435 BR SBDONE ; YES, DONE.
1171
1172 ; WE'RE HERE IF TEST INSTRUCTION TRAPPED.
1173
1174 011316 011637 001164 SB04: MOV (SP), $TMP2 ; GET TRAP PC.
1175 011322 022626 CMP (SP)+, (SP)+
1176 011324 104047 ERROR 47
1177 011326 000430 BR SBDONE
1178
1179 011330 000177 167574 137271 SB10: .WORD 177,167574,137271,107675 ; EXPONENT = 0.
1180 011340 000000 000000 000000 SB11: .WORD 0,0,0,0 ; EXPECTED RESULT.
1181
1182 011350 123245 026720 122324 SB12: .WORD 123245,26720,122324,52672 ; NEG, EXPON NON-ZERO.
1183 011360 023245 026720 122324 SB13: .WORD 023245,26720,122324,52672 ; EXPECTED RESULT.
1184
1185 011370 177777 177777 177777 .WORD -1,-1,-1,-1 ; IN CASE IT AUTO-DECR.
1186 011400 000000 000000 000000 SB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1187
1188 011410 SBDONE:
(1) 011410 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 011412 000400 BR TST2 ;: ...AND PROCEED.
```

```
1190 .....  
(3) *TEST 22 SPECIAL FDST FLOW, USING ABSD MODE 4  
(4) *  
(4) * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST  
(4) * INSTRUCTIONS, USING ABSD, MODE 4.  
(4) * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,  
(4) * AND AGAIN WITH E(FDST) NON-ZERO.  
(3) .....  
(2) 011414 000004 TST22: SCOPE  
1191 :  
1192 : THIS COMBINES TESTS 23 AND 35 FROM FP11-A, PART 3.  
1193 :  
1194 011416 012737 011656 001172 TB1: MOV #TB10,$TMP5 ; INITIAL DATA (EXP 0).  
1195 011424 012737 011666 001174 MOV #TB11,$TMP6 ; EXPECTED RESULT.  
1196 011432 012737 000204 001170 MOV #204,$TMP4 ; AND FPS.  
1197 011440 000411 BR TB3  
1198 011442 012737 011676 001172 TB2: MOV #TB12,$TMP5 ; INITIAL DATA (EXP NON-ZERO).  
1199 011450 012737 011706 001174 MOV #TB13,$TMP6 ; EXPECTED RESULT.  
1200 011456 012737 000200 001170 MOV #200,$TMP4 ; AND FPS.  
1201 :  
1202 011464 TB3: :  
(1) 011464 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
1203 011466 170011 SETD  
1204 011470 013700 001172 MOV $TMP5,R0  
1205 011474 012701 011716 MOV #TB14,R1  
1206 011500 012702 000004 MOV #4,R2  
1207 011504 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.  
1208 011506 077202 SOB R2,1$  
1209 :  
1210 011510 012737 011644 000004 MOV #TB04,ERRVEC ; SET BUS-ERR TRAP.  
1211 011516 012737 011534 001164 MOV #2$,$TMP2  
1212 011524 012700 011726 MOV #TB14+10,R0 ; SET FDST.  
1213 011530 010037 001166 MOV R0,$TMP3 ; SAVE AS BEFORE VALUE.  
1214 011534 170640 2$: ABSD -(R0) ; TEST INSTRUCTION.  
1215 011536 000241 241  
1216 011540 020027 011716 CMP R0,#TB14 ; AUTO-DECR OK ??  
1217 011544 001407 BEQ 3$ ; BR IF SO.  
1218 011546 012737 011716 001170 MOV #TB14,$TMP4  
1219 011554 010037 001172 MOV R0,$TMP5  
1220 011560 104050 ERROR 50 ; R0 INCORRECT.  
1221 011562 000465 BR TBDONE  
1222 :  
1223 011564 170200 3$: STFPS R0 ; GET FPS.  
1224 011566 010037 001166 MOV R0,$TMP3  
1225 011572 013737 001174 011604 MOV $TMP6,4$ ; SET EXPECTED POINTER...  
1226 011600 004537 004062 JSR R5,CHECK4 ; ...AND CHECK RESULT.  
1227 011604 000000 011716 4$: 0,TB14  
1228 011610 001004 BNE 5$ ; BR IF WRONG.  
1229 011612 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK STATUS.  
1230 011620 001404 BEQ 6$ ; BR IF BOTH OK.  
1231 011622 012737 011716 001176 5$: MOV #TB14,$TMP7  
1232 011630 104051 ERROR 51 ; DATA OR FPS WRONG.  
1233 :  
1234 011632 023727 001170 000200 6$: CMP $TMP4,#200 ; 2ND PASS ??  
1235 011640 001300 BNE TB2 ; NO, DO IT NOW.
```

```
1236 011642 000435 BR TBDONE ; YES, ALL DONE.
1237
1238 ; TEST INSTRUCTION TRAPPED TO 4.
1239
1240 011644 011637 001164 TB04: MOV (SP), $TMP2
1241 011650 022626 CMP (SP)+, (SP)+
1242 011652 104052 ERROR 52
1243 011654 000430 BR TBDONE
1244
1245 011656 000177 117273 147576 TB10: .WORD 177,117273,147576,177071 ; EXPONENT = 0.
1246 011666 000000 000000 000000 TB11: .WORD 0,0,0,0 ; EXPECTED RESULT.
1247
1248 011676 123245 026720 122324 TB12: .WORD 123245,26720,122324,52672 ; NEG, EXPN NON-ZERO.
1249 011706 023245 026720 122324 TB13: .WORD 023245,26720,122324,52672 ; EXPD RESULT.
1250
1251 011716 000000 000000 000000 TB14: .WORD 0,0,0,0 ; DATA TO TEST.
1252 011726 177777 177777 177777 .WORD -1,-1,-1,-1 ; IN CASE AUTO DECR BAD.
1253
1254 011736 TBDONE:
(1) 011736 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 011740 000400 BR TST23 ;:...AND PROCEED.
```

```

1256 (3) *****
      (4) : TEST 23 SPECIAL FDST FLOW, USING NEGD MODE 3
      (4) :
      (4) : * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
      (4) : * INSTRUCTIONS, USING NEGD, MODE 3.
      (4) : * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) 0,
      (4) : * AND AGAIN WITH E(FDST) NON-ZERO.
      (3) : *****
      (2) 011742 000004 TST23: SCOPE
1257 :
1258 : THIS COMBINES TESTS 24 AND 36 FROM FP11-A, PART 3.
1259 :
1260 011744 012737 012204 001172 UB1: MOV #UB10,$TMP5 ; INITIAL DATA (EXPN - 0).
1261 011752 012737 012214 001174 MOV #UB11,$TMP6 ; EXPECTED RESULT.
1262 011760 012737 000204 001170 MOV #204,$TMP4 ; AND FPS.
1263 011766 000411 BR UB3
1264 011770 012737 012224 001172 UB2: MOV #UB12,$TMP5 ; INITIAL DATA (POS NON-ZERO).
1265 011776 012737 012234 001174 MOV #UB13,$TMP6 ; EXPD RESULT.
1266 012004 012737 000210 001170 MOV #210,$TMP4 ; AND FPS.
1267 :
1268 012012 UB3:
      (1) 012012 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
1269 012014 170011 SETD
1270 012016 013700 001172 MOV $TMP5,R0
1271 012022 012701 012244 MOV #UB14,R1
1272 012026 012702 000004 MOV #4,R2
1273 012032 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
1274 012034 077202 SOB R2,1$
1275 :
1276 012036 012737 012172 000004 MOV #UB04,ERRVEC ; SET BUS-ERR TRAP.
1277 012044 012737 012062 001164 MOV #25,$TMP2
1278 012052 012700 012262 MOV #UB15,R0 ; SET UP FDST.
1279 012056 010037 001166 MOV R0,$TMP3 ; SAVE
1280 012062 170730 2$: NEGD @ (R0)+ ; TEST INSTRUCTION.
1281 012064 000241 241
1282 012066 020027 012264 CMP R0,#UB15+2 ; AUTO-INCR OK ??
1283 012072 001407 BEQ 3$
1284 012074 012737 012264 001170 MOV #UB15+2,$TMP4
1285 012102 010037 001172 MOV R0,$TMP5
1286 012106 104053 ERROR 53 ; R0 WRONG.
1287 012110 000470 BR UBDONE
1288 :
1289 012112 170200 3$: STFPS R0 ; GET FPS.
1290 012114 010037 001166 MOV R0,$TMP3
1291 012120 013737 001174 012132 MOV $TMP6,4$ ; SET EXPD DATA POINTER.
1292 012126 004537 004062 JSR R5,CHECK4 ; AND CHECK DATA.
1293 012132 000000 012244 4$: O,UB14
1294 012136 001004 BNE 5$ ; BR IF WRONG.
1295 012140 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK STATUS.
1296 012146 001404 BEQ 6$ ; BR IF BOTH WERE OK.
1297 012150 012737 012244 001176 5$: MOV #UB14,$TMP7
1298 012156 104054 ERROR 54 ; DATA OR FPS WRONG.
1299 :
1300 012160 023727 001170 000210 6$: CMP $TMP4,#210 ; 2ND PASS ??
1301 012166 001300 BNE UB2 ; NO, DO IT.

```

```
1302 012170 000440 BR UBDONE ; YES.
1303
1304 :: TEST INSTR TRAPPED TO 4.
1305
1306 012172 011637 001164 UB04: MOV (SP), $TMP2
1307 012176 022626 CMP (SP)+, (SP)+
1308 012200 104055 ERROR 55
1309 012202 000433 BR UBDONE
1310
1311 012204 000177 147576 177071 UB10: .WORD 177,147576,177071,107576 ; EXPON = 0.
1312 012214 000000 000000 000000 UB11: .WORD 0,0,0,0 ; EXPD RESULT.
1313
1314 012224 023245 026720 122324 UB12: .WORD 023245,26720,122324,52672 ; POS NON-ZERO.
1315 012234 123245 026720 122324 UB13: .WORD 123245,26720,122324,52672 ; EXPD RESULT.
1316
1317 012244 000000 000000 000000 UB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1318
1319 012254 177777 177777 177777 .WORD -1,-1,-1
1320 012262 012244 UB15: .WORD UB14 ; DST POINTER.
1321 012264 177777 177777 177777 .WORD -1,-1,-1
1322
1323 012272 UBDONE:
(1) 012272 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 012274 000400 BR TST24 ;: ...AND PROCEED.
```

```
1325 .....  
(3) : *TEST 24 SPECIAL FDST FLOW, USING NEG D MODE 5  
(4) :  
(4) : * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND 1ST  
(4) : * INSTRUCTIONS, USING NEG D, MODE 5.  
(4) : * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,  
(4) : * AND AGAIN WITH E(FDST) NON-ZERO.  
(3) :  
(2) 012276 000004 : .....  
1326 : TST24: SCOPE  
1327 : THIS COMBINES TESTS 25 AND 37 FROM FP11-A, PART 3.  
1328 :  
1329 012300 012737 012540 001172 VB1: MOV #VB10,$TMP5 ; INITIAL DATA (EXPN 0).  
1330 012306 012737 012550 001174 MOV #VB11,$TMP6 ; EXPECTED RESULT.  
1331 012314 012737 000204 001170 MOV #204,$TMP4 ; AND FPS.  
1332 012322 000411 BR VB3  
1333 012324 012737 012560 001172 VB2: MOV #VB12,$TMP5 ; INITIAL DATA (POS NON-ZERO).  
1334 012332 012737 012570 001174 MOV #VB13,$TMP6  
1335 012340 012737 000210 001170 MOV #210,$TMP4  
1336 :  
1337 012346 VB3: LUPERR ;  
(1) 012346 104411 SETD ; LOOP HERE ON ERROR IF SWR9 - 1.  
1338 012350 170011 MOV $TMP5,R0  
1339 012352 013700 001172 MOV #VB14,R1  
1340 012356 012701 012600 MOV #4,R2  
1341 012362 012702 000004 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.  
1342 012366 012021 SOB R2,1$  
1343 012370 077202  
1344 :  
1345 012372 012737 012526 000004 MOV #VB04,ERRVEC ; IN CASE IT TRAPS.  
1346 012400 012737 012416 001164 MOV #2$, $TMP2  
1347 012406 012700 012620 MOV #VB15+2,R0 ; SET UP FDST.  
1348 012412 010037 001166 MOV R0,$TMP3 ; SAVE IT.  
1349 012416 170750 2$: NEGD @-(R0) ; TEST INSTRUCTION.  
1350 012420 000241 241  
1351 012422 020027 012616 CMP R0,#VB15 ; AUTO-DECR OK ??  
1352 012426 001407 BEQ 3$ ; BR IF SO.  
1353 012430 012737 012616 001170 MOV #VB15,$TMP4  
1354 012436 010037 001172 MOV R0,$TMP5  
1355 012442 104056 ERROR 56 ; R0 INCORRECT.  
1356 012444 000471 BR VBDONE  
1357 :  
1358 012446 170200 3$: STFPS R0 ; GET FPS.  
1359 012450 010037 001166 MOV R0,$TMP3  
1360 012454 013737 001174 012466 MOV $TMP6,4$ ; SET POINTER.  
1361 012462 004537 004062 JSR R5,CHECK4 ; AND CHECK DATA.  
1362 012466 000000 012600 4$: 0,VB14  
1363 012472 001004 BNE 5$ ; ER IF WRONG.  
1364 012474 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK FPS.  
1365 012502 001404 BEQ 6$ ; BR IF BOTH WERE OK.  
1366 012504 012737 012600 001176 5$: MOV #VB14,$TMP7  
1367 012512 104057 ERROR 57 ; DATA OR FPS WRONG.  
1368 :  
1369 012514 023727 001170 000210 6$: CMP $TMP4,#210 ; 2ND PASS ??  
1370 012522 001300 BNE VB2 ; NO, DO IT NOW.
```

```
1371 012524 000441 BR VBDONE ; AND QUIT.
1372
1373 ; TEST INSTRUCTION TRAPPED TO 4.
1374
1375 012526 011637 001164 VB04: MOV (SP), $TMP2
1376 012532 022626 CMP (SP)+, (SP)+
1377 012534 104060 ERROR 60
1378 012536 000434 BR VBDONE
1379
1380 012540 000176 177074 127374 VB10: .WORD 176,177074,127374,157677 ; EXPON = 0.
1381 012550 000000 000000 000000 VB11: .WORD 0,0,0,0 ; EXPECTED RESULT.
1382
1383 012560 023245 026720 122324 VB12: .WORD 023245,26720,122324,52672 ; POS NON-ZERO.
1384 012570 123245 026720 122324 VB13: .WORD 123245,26720,122324,52672 ; EXPECTED RESULT.
1385
1386 012600 000000 000000 000000 VB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1387
1388 012610 177777 177777 177777 .WORD -1,-1,-1
1389 012616 012600 VB15: .WORD VB14 ; FDST POINTER.
1390 012620 177777 177777 177777 .WORD -1,-1,-1,-1
1391
1392 012630 VBDONE:
(1) 012630 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 012632 000400 BR TST25 ;: ...AND PROCEED.
```

```

1394 .....
(3) *TEST 25 SPECIAL FDST FLOW, USING ABSD MODE 6
(4) *
(4) * TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TST
(4) * INSTRUCTIONS, USING ABSD, MODE 6.
(4) * EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
(4) * AND AGAIN WITH E(FDST) NON-ZERO.
(4) *
(3) .....
(2) 012634 000004 TST25: SCOPE
1395 :
1396 : THIS COMBINES TESTS 26 AND 42 FROM FP11-A, PART 3.
1397 :
1398 012636 012737 013100 001172 WB1: MOV #WB10,$TMP5 ; INITIAL DATA (EXPON 0).
1399 012644 012737 013110 001174 MOV #WB11,$TMP6 ; EXPD RESULT.
1400 012652 012737 000204 001170 MOV #204,$TMP4 ; AND FPS.
1401 012660 000411 BR WB3
1402 012662 012737 013120 001172 WB2: MOV #WB12,$TMP5 ; INITIAL (NEG NON-ZERO).
1403 012670 012737 013130 001174 MOV #WB13,$TMP6
1404 012676 012737 000200 001170 MOV #200,$TMP4
1405
1406 012704 WB3:
(1) 012704 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
1407 012706 170011 SETD
1408 012710 013700 001172 MOV $TMP5,R0
1409 012714 012701 013140 MOV #WB14,R1
1410 012720 012702 000004 MOV #4,R2
1411 012724 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
1412 012726 077202 SOB R2,1$
1413
1414 012730 012737 013066 000004 MOV #WB04,ERRVEC ; IN CASE IT TRAPS.
1415 012736 012737 012754 001164 MOV #2$,$TMP2
1416 012744 012700 012677 MOV #WB14-241,R0 ; SET UP THE OPERAND ADDRESS.
1417 012750 010037 001166 MOV R0,$TMP3 ; SAVE IT.
1418 012754 170660 000241 2$: ABSD 241(R0) ; TEST INSTRUCTION.
1419 012760 000241 241
1420 012762 020037 001166 CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.
1421 012766 001407 BEQ 3$ ; BR IF SO.
1422 012770 013737 001166 001170 MOV $TMP3,$TMP4
1423 012776 010037 001172 MOV R0,$TMP5
1424 013002 104061 ERROR 61 ; R0 BAD.
1425 013004 000461 BR WBDONE
1426
1427 013006 170200 3$: STFPS R0 ; GET FPS.
1428 013010 010037 001166 MOV R0,$TMP3
1429 013014 013737 001174 013026 MOV $TMP6,4$ ; SET POINTER.
1430 013022 004537 004062 JSR R5,CHECK4 ; AND CHECK RESULT.
1431 013026 000000 013140 4$: 0,WB14
1432 013032 001004 BNE 5$ ; BR IF WRONG.
1433 013034 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK FPS.
1434 013042 001404 BEQ 6$ ; BR IF BOTH OK.
1435 013044 012737 013140 001176 5$: MOV #WB14,$TMP7
1436 013052 104062 ERROR 62
1437
1438 013054 023727 001170 000200. 6$: CMP $TMP4,#200 ; 2ND PASS ??
1439 013062 001277 BNE WB2 ; NO, GO 'ROUND.
    
```



```
1440 013064 000431 BR WBDONE
1441
1442 :: TEST INSTRUCTION TRAPPED TO 4.
1443 ::
1444 013066 011637 001164 WB04: MOV (SP), $TMP2
1445 013072 022626 CMP (SP)+, (SP)+
1446 013074 104063 ERROR 63
1447 013076 000424 BR WBDONE
1448
1449 013100 000177 161524 131273 WB10: .WORD 177,161524,131273,107174 ; EXPON = 0.
1450 013110 000000 000000 000000 WB11: .WORD 0,0,0,0 ; EXPECTED RESULT.
1451
1452 013120 123245 026720 122324 WB12: .WORD 123245,26720,122324,52672 ; NEG NON-ZERO.
1453 013130 023245 026720 122324 WB13: .WORD 023245,26720,122324,52672 ; EXPECTED.
1454
1455 013140 000000 000000 000000 WB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1456
1457 WBDONE:
(1) 013150 104412 CLRFP
(3) 013152 000400 BR TST26 ;: CLEAR FP STATUS...
;: ...AND PROCEED.
```

```

1459
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 013154 000004
1460
1461
1462
1463 013156 012737 013420 001172
1464 013164 012737 013430 001174
1465 013172 012737 000204 001170
1466 013200 000411
1467 013202 012737 013440 001172
1468 013210 012737 013450 001174
1469 013216 012737 000200 001170
1470
1471 013224
(1) 013224 104411
1472 013226 170011
1473 013230 013700 001172
1474 013234 012701 013460
1475 013240 012702 000004
1476 013244 012021
1477 013246 077202
1478
1479 013250 012737 013406 000004
1480 013256 012737 013274 001164
1481 013264 012700 013227
1482 013270 010037 001166
1483 013274 170670 000241
1484 013300 000241
1485 013302 020037 001166
1486 013306 001407
1487 013310 013737 001166 001170
1488 013316 010037 001172
1489 013322 104064
1490 013324 000462
1491
1492 013326 170200
1493 013330 010037 001166
1494 013334 013737 001174 013346
1495 013342 004537 004062
1496 013346 000000 013460
1497 013352 001004
1498 013354 023737 001166 001170
1499 013362 001404
1500 013364 012737 013460 001176
1501 013372 104065
1502
1503 013374 023727 001170 000200
1504 013402 001277

```

```

*****
*TEST 26 SPECIAL FDST FLOW, USING ABSD MODE 7
*
* TEST THE SPECIAL FDST MODE USED BY THE ABS, NEG, AND TSI
* INSTRUCTIONS, USING ABSD, MODE 7.
* EXECUTE THE TEST TWICE, FIRST WITH E(FDST) = 0,
* AND AGAIN WITH E(FDST) NON-ZERO.
*****
TST26: SCOPE
; THIS COMBINES TESTS 27 AND 43 FROM FP11-A, PART 3.
YB1: MOV #YB10,$TMP5 ; INITIAL DATA (EXPON 0).
MOV #YB11,$TMP6 ; EXPECTED RESULT.
MOV #204,$TMP4 ; AND FPS.
BR YB3
YB2: MOV #YB12,$TMP5 ; INITIAL DATA (NEG NON-ZERO).
MOV #YB13,$TMP6
MOV #200,$TMP4
YB3: LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
SETD
MOV $TMP5,R0
MOV #YB14,R1
MOV #4,R2
1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
SOB R2,1$
MOV #YB04,ERRVEC ; IN CASE IT TRAPS.
MOV #2$,$TMP2
MOV #YB15-241,R0 ;SET UP THE OPERAND ADDRESS.
MOV R0,$TMP3 ; SAVE R0 BEFORE.
2$: ABSD @241(R0) ;TEST INSTRUCTION.
241
CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.
BEQ 3$ ; BR IF SO.
MOV $TMP3,$TMP4
MOV R0,$TMP5
MOV ERROR 64 ; R0 BAD.
BR YBDONE
3$: STFPS R0 ;GET FPS.
MOV R0,$TMP3
MOV $TMP6,4$ ; SET EXPD POINTER.
JSR R5,CHECK4 ; AND CHECK RESULT.
4$: MOV 0,YB14
BNE 5$ ; BR IF WRONG.
CMP $TMP3,$TMP4 ; CHECK FPS.
BEQ 6$ ; BR IF BOTH OK.
5$: MOV #YB14,$TMP7
MOV ERROR 65
6$: CMP $TMP4,#200 ; 2ND PASS ??
BNE YB2 ; NO

```

```
1505 013404 000432 BR YBDONE ; YES.
1506
1507 ; TEST INSTRUCTION TRAPPED.
1508 ;
1509 013406 011637 001164 YB04: MOV (SP), $TMP2
1510 013412 022626 CMP (SP)+, (SP)+
1511 013414 104066 ERROR 66
1512 013416 000425 BR YBDONE
1513
1514 013420 000177 167574 137271 YB10: .WORD 177,167574,137271,107675 ; EXPON - 0.
1515 013430 000000 000000 000000 YB11: .WORD 0,0,0,0 ; EXPECT RESULT.
1516
1517 013440 123245 026720 123324 YB12: .WORD 123245,26720,123324,52672 ; NEG NON-ZERO.
1518 013450 023245 026720 123324 YB13: .WORD 023245,26720,123324,52672 ; EXPECT RESULT.
1519
1520 013460 000000 000000 000000 YB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1521
1522 013470 013460 YB15: YB14 ; INDIRECT POINTER.
1523
1524 013472 YBDONE:
(1) 013472 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 013474 000400 BR TST27 ;: ...AND PROCEED.
```

```
1532 (3) *****  
(4) *TEST 27 SPECIAL FDST FLOW, USING NEGD MODE 6 WITH GR7  
(4) *  
(4) * TEST SPECIAL FDST FLOW, MODE 6, WITH GR7 (PC RELATIVE),  
(4) * USING THE NEGD INSTRUCTION.  
(3) *****  
(2) 013476 000004 TST27: SCOPE  
1533  
1534 013500 012737 013706 001172 ZB1: MOV #ZB10,$TMP5 ; INITIAL DATA (EXPON = 0).  
1535 013506 012737 013716 001174 MOV #ZB11,$TMP6 ; EXPD RESULT.  
1536 013514 012737 000204 001170 MOV #204,$TMP4 ; AND FPS.  
1537 013522 000411 BR ZB3  
1538 013524 012737 013726 001172 ZB2: MOV #ZB12,$TMP5 ; INITIAL DATA (POS NON-ZERO).  
1539 013532 012737 013736 001174 MOV #ZB13,$TMP6  
1540 013540 012737 000210 001170 MOV #210,$TMP4  
1541  
1542 013546 ZB3: LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
(1) 013546 104411 SETD  
1543 013550 170011 MOV $TMP5,R0  
1544 013552 013700 001172 MOV #ZB14,R1  
1545 013556 012701 013746 MOV #4,R2  
1546 013562 012702 000004 1$: MOV (R0)+,(R1)+ ; SET INIIAL DATA.  
1547 013566 012021 SOB R2,1$  
1548 013570 077202  
1549  
1550 013572 012737 013674 000004 MOV #ZB04,ERRVEC ; SET TRAP.  
1551 013600 012737 013606 001164 MOV #2$, $TMP2  
1552 .DSABL AMA ; *** TO INSURE MODE 67 ***  
1553 013606 170767 000134 2$: NEGD ZB14 ; TEST INSTRUCTION.  
1554 .ENABL AMA  
1555 013612 000241 241  
1556 013614 170205 STFPS R5 ; GET FPS.  
1557 013616 010537 001166 MOV R5,$TMP3  
1558 013622 013737 001174 013634 MOV $TMP6,3$ ; SET POINTER.  
1559 013630 004537 004062 JSR R5,CHECK4 ; AND CHECK DATA.  
1560 013634 000000 013746 3$: 0,ZB14  
1561 013640 001004 BNE 4$ ; BR IF WRONG.  
1562 013642 023737 001166 001170 CMP $TMP3,$TMP4 ; CHECK FPS.  
1563 013650 001404 BEQ 5$ ; BR IF BOTH OK.  
1564 013652 012737 013746 001176 4$: MOV #ZB14,$TMP7  
1565 013660 104067 ERROR 67 ; DATA OR FPS WRONG.  
1566  
1567 013662 023727 001170 000210 5$: CMP $TMP4,#210 ; 2ND PASS ??  
1568 013670 001315 BNE ZB2 ; NO.  
1569 013672 000431 BR ZBDONE ; YES.  
1570  
1571 ;: TEST INSTRUCTION TRAPPED.  
1572  
1573 013674 011637 001164 ZB04: MOV (SP),$TMP2  
1574 013700 022626 CMP (SP)+,(SP)+  
1575 013702 104070 ERROR 70  
1576 013704 000424 BR ZBDONE  
1577  
1578 013706 000127 137475 147372 ZB10: .WORD 127,137475,147372,117057 ; EXPON = 0.  
1579 013716 000000 000000 000000 ZB11: .WORD 0,0,0,0 ; EXPECT.
```

1580
1581 013726 011127 137475 147372 ZB12: .WORD 011127,137475,147372,117057 ; POS NON-ZERO.
1582 013736 111127 137475 147372 ZB13: .WORD 111127,137475,147372,117057 ; EXPECT.
1583
1584 013746 000000 000000 000000 ZB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1585
1586 013756 ZBDONE:
(1) 013756 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 013760 000400 BR TST30 ;:...AND PROCEED.

```

1594 (3) *****
      (4) *TEST 30 SPECIAL FDST FLOW, USING ABSD MODE 7 WITH GR7
      (4) *
      (4) * TEST SPECIAL FDST FLOW, MODE 7, WITH GR7 (PC RELATIVE DEFERRED,
      (4) * USING THE ABSD INSTRUCTION.
      (3) *****
      (2) 013762 000004 TST30: SCOPE
1595
1596 013764 012737 014172 001172 AAB1: MOV #AAB10,$TMP5 ; INITIAL DATA (EXP = 0).
1597 013772 012737 014202 001174 MOV #AAB11,$TMP6 ; EXPECT.
1598 014000 012737 000204 001170 MOV #204,$TMP4 ; FPS.
1599 014006 000411 BR AAB3
1600 014010 012737 014212 001172 AAB2: MOV #AAB12,$TMP5 ; INITIAL DATA (NEG NON-ZERO).
1601 014016 012737 014222 001174 MOV #AAB13,$TMP6
1602 014024 012737 000200 001170 MOV #200,$TMP4
1603
1604 014032 AAB3:
      (1) 014032 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
1605 014034 170011 SETD
1606 014036 013700 001172 MOV $TMP5,R0
1607 014042 012701 014232 MOV #AAB14,R1
1608 014046 012702 000004 MOV #4,R2
1609 014052 012021 1$: MOV (R0)+,(R1)+ ; SET INITIAL DATA.
1610 014054 077202 SOB R2,1$
1611
1612 014056 012737 014160 000004 MOV #AAB04,ERRVEC ; IN CASE IT TRAPS.
1613 014064 012737 014072 001164 MOV #2$, $TMP2
1614 .DSABL AMA ; *** INSURE MODE 77 ****
1615 014072 170677 000144 2$: ABSD @AAB15 ; TEST INSTRUCTION.
1616 .ENABL AMA
1617 014076 000241 241
1618 014100 170205 STFPS R5 ; GET FPS.
1619 014102 010537 001166 MOV R5,$TMP3
1620 014106 013737 001174 014120 MOV $TMP6,3$ ; SET POINTER.
1621 014114 004537 004062 JSR R5,CHECK4 ; AND CHECK RESULT.
1622 014120 000000 014232 3$: 0,AAB14
1623 014124 001004 BNE 4$ ; BR IF WRONG.
1624 014126 023737 001166 001170 CMP $TMP3,$TMP4
1625 014134 001404 BEQ 5$ ; BR IF BOTH OK.
1626 014136 012737 014232 001176 4$: MOV #AAB14,$TMP7
1627 014144 104071 ERROR 71
1628
1629 014146 023727 001170 000200 5$: CMP $TMP4,#200 ; 2ND PASS ??
1630 014154 001315 BNE AAB2 ; NO.
1631 014156 000432 BR AABDONE ; YES.
1632
1633 ; TEST INSTRUCTION TRAPPED.
1634
1635 014160 011637 001164 AAB04: MOV (SP),$TMP2
1636 014164 022626 CMP (SP)+,(SP)+
1637 014166 104072 ERROR 72
1638 014170 000425 BR AABDONE
1639
1640 014172 100137 045607 101230 AAB10: .WORD 100137,45607,101230,45607 ; EXP = 0.
1641 014202 000000 000000 000000 AAB11: .WORD 0,0,0,0 ; EXPECT.

```

```

1642
1643 014212 123137 045607 101230 AAB12: .WORD 123137,45607,101230,45607 ; NEG NON-ZERO.
1644 014222 023137 045607 101230 AAB13: .WORD 023137,45607,101230,45607 ; EXPECT.
1645
1646 014232 000000 000000 000000 AAB14: .WORD 0,0,0,0 ; DATA UNDER TEST.
1647
1648 014242 014232 AAB15: .WORD AAB14 ; INDIRECT POINTER.
1649
1650 014244 AABDONE:
(1) 014244 104412 CLRFP5 ;: CLEAR FP STATUS...
(3) 014246 000400 BR TST31 ;:...AND PROCEED.

```

```

1658 .....
(3) *TEST 31 SPECIAL FDST FLOW, USING NEGD MODE 2 WITH GR7
(4) *
(4) * TEST SPECIAL FDST FLOW, MODE 2, WITH GR7 (PC IMMEDIATE),
(4) * USING THE NEGD INSTRUCTION.
(3) .....
(2) 014250 000004 TST31: SCOPE
1659 014252 BBB1:
(1) 014252 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
1660 014254 170011 SETD
1661 014256 012700 014442 MOV #BBB10,R0
1662 014262 012701 014320 MOV #BBB12,R1
1663 014266 012702 000004 MOV #4,R2
1664 014272 012021 1$: MOV (R0)+,(R1)+ ; SET UP DATA BUFFER.
1665 014274 077202 SOB R2,1$
1666
1667 014276 012737 014316 001164 MOV #2$,$TMP2
1668 014304 012737 000200 001170 MOV #200,$TMP4 ; EXPD FPS.
1669 014312 012701 014330 MOV #2$+12,R1 ; TO CALCULATE PC AFTER.
1670 014316 170727 2$: NEGD (PC)+ ; TEST INSTRUCTION.
1671 014320 105741 BBB12: 105741 ; SHOULD CHANGE TO 5741.
1672 014322 005741 TST -(R1) ; THESE SHOULD EXECUTE...
1673 014324 005741 TST -(R1) ; ...AND YIELD RETURN PC...
1674 014326 005741 TST -(R1) ; ...IN R1.
1675
1676 014330 170205 STFPS R5 ; GET FPS.
1677 014332 010537 001166 MOV R5,$TMP3
1678 014336 004537 004062 JSR R5,CHECK4 ; CHECK DATA.
1679 014342 014452 014320 BBB11, BBB12
1680 014346 001004 BNE 1$ ; BR IF WRONG.
1681 014350 023737 001166 001170 CMP $TMP3,$TMP4
1682 014356 001412 BEQ 2$ ; BR IF BOTH OK.
1683 014360 012737 014442 001172 1$: MOV #BBB10,$TMP5
1684 014366 012737 014452 001174 MOV #BBB11,$TMP6
1685 014374 012737 014320 001176 MOV #BBB12,$TMP7
1686 014402 104073 ERROR 73 ; DATA OR FPS WRONG.
1687
1688 014404 020127 014322 2$: CMP R1,#BBB12+2 ; WAS FINAL PC RIGHT ??
1689 014410 001406 BEQ 3$ ; YUP.
1690 014412 010137 001166 MOV R1,$TMP3
1691 014416 012737 014322 001170 MOV #BBB12+2,$TMP4
1692 014424 104074 ERROR 74 ; PC BAD.
1693 014426 000415 3$: BR BBBDONE
1694
1695 ; TEST INSTR TRAPPED.
1696
1697 014430 011637 001164 BBB04: MOV (SP),$TMP2
1698 014434 022626 CMP (SP)+,(SP)+
1699 014436 104075 ERROR 75
1700 014440 000410 BR BBBDONE
1701
1702 014442 105741 005741 005741 BBB10: 105741,5741,5741,5741 ; 4 TST -(R1)'S (1ST ONE NEG).
1703 014452 005741 005741 005741 BBB11: 005741,5741,5741,5741 ; EXPECT RESULT.
1704
1705 014462 BBBDONE:
    
```


CFPBA -- LSI11/23 FPF11 DIAGNOSTIC, PART 2
CFPBA.P11 12-FEB-81 10:27 T31

H 6
MACY11 30G(1063) 12-FEB-81 11:04 PAGE 2-42
SPECIAL FDST FLOW, USING NEGD MODE 2 WITH GR7

SEQ 0072

(1) 014462 104412
(3) 014464 000400

CLRFPS
BR TST32
:: CLEAR FP STATUS...
::...AND PROCEED.

```
1712 .....  
  (3) *TEST 32      NEG, ABSD, AND TSTD -- FDST MODE 1  
  (4) *  
  (4) * TEST NEG, ABSD, AND TSTD, WITH VARIOUS OPERANDS.  
  (4) *  
  (3) .....  
  (2) 014466 000004 TST32: SCOPE  
1713  
1714 ;TEST 'NEG' WITH POS NON-ZERO OPERAND.  
1715  
1716 014470 004737 015046 CCB1: JSR      PC,CCB10          ; NEG ENTRY.  
1717 014474 016341 055772 021133 .WORD 016341,55772,21133,55447 ; OPERAND.  
1718 014504 116341 055772 021133 .WORD 116341,55772,21133,55447 ; RESULT.  
1719 014514 000217          .WORD 217          ; FPS BEFORE.  
1720 014516 000210 000000 .WORD 210,0        ; FPS, FEC AFTER.  
1721  
1722 ;TEST 'NEG' WITH NEG OPERAND.  
1723  
1724 014522 004737 015046 CCB2: JSR      PC,CCB10          ;  
1725 014526 152525 053545 055565 .WORD 152525,53545,55565,57505 ; OPERAND.  
1726 014536 052525 053545 055565 .WORD 052525,53545,55565,57505 ; RESULT.  
1727 014546 000217          .WORD 217          ; FPS BEFORE.  
1728 014550 000200 000000 .WORD 200,0        ; FPS, FEC AFTER.  
1729  
1730 ;TEST 'ABSD' WITH POSITIVE OPERAND.  
1731  
1732 014554 004737 015064 CCB3: JSR      PC,CCB20          ; ABSD ENTRY.  
1733 014560 060705 124735 060124 .WORD 060705,124735,60124,73560 ; OPERAND.  
1734 014570 060705 124735 060124 .WORD 060705,124735,60124,73560 ; RESULT.  
1735 014600 000217          .WORD 217          ; FPS BEFORE.  
1736 014602 000200 000000 .WORD 200,0        ; FPS, FEC AFTER.  
1737  
1738 ;TEST 'ABSD' WITH NEGATIVE OPERAND.  
1739  
1740 014606 004737 015064 CCB4: JSR      PC,CCB20          ;  
1741 014612 154345 076567 032123 .WORD 154345,76567,32123,43234 ; OPERAND.  
1742 014622 054345 076567 032123 .WORD 054345,76567,32123,43234 ; RESULT.  
1743 014632 000217          .WORD 217          ; FPS  
1744 014634 000200 000000 .WORD 200,0        ; FPS, FEC AFTER.  
1745  
1746 ;TEST 'TSTD' WITH POSITIVE NON-ZERO OPERAND.  
1747  
1748 014640 004737 015102 CCB5: JSR      PC,CCB30          ; TSTD ENTRY.  
1749 014644 012321 045654 070107 .WORD 012321,45654,70107,34543 ; OPERAND.  
1750 014654 012321 045654 070107 .WORD 012321,45654,70107,34543 ; RESULT.  
1751 014664 000217          .WORD 217          ; FPS  
1752 014666 000200 000000 .WORD 200,0        ; FPS, FEC AFTER.  
1753  
1754 ;TEST 'TSTD' WITH NEG NON-ZERO OPERAND.  
1755  
1756 014672 004737 015102 CCB6: JSR      PC,CCB30          ;  
1757 014676 123765 023407 034510 .WORD 123765,23407,34510,45621 ; OPERAND.  
1758 014706 123765 023407 034510 .WORD 123765,23407,34510,45621 ; RESULT  
1759 014716 000217          .WORD 217          ; FPS  
1760 014720 000210 000000 .WORD 210,0        ; FPS, FEC AFTER.  
1761 ;
```

```

1762 ;TEST 'TSTD' WITH E(OPERAND) = 0.
1763
1764 014724 004737 015102 CCB7: JSR PC,CCB30
1765 014730 000175 176737 071727 .WORD 000175,176737,71727,37574 ; OPERAND.
1766 014740 000175 176737 071727 .WORD 000175,176737,71727,37574 ; RESULT
1767 014750 000217 .WORD 217 ; FPS
1768 014752 000204 000000 .WORD 204,0 ; FPS, FEC AFTER.
1769
1770 ; TEST 'TSTD' WITH A NEG ZERO (-0) AND FIUV = 0.
1771
1772 014756 004737 015102 CCB8: JSR PC,CCB30
1773 014762 100123 021012 034565 .WORD 100123,21012,34565,43210 ; OPERAND.
1774 014772 100123 021012 034565 .WORD 100123,21012,34565,43210 ; RESULT
1775 015002 000217 .WORD 217 ; FPS (FIUV = 0).
1776 015004 000214 000000 .WORD 214,0 ; FPS, FEC AFTER.
1777
1778 ; TEST 'TSTD' WITH NEG ZERO (-0) AND FIUV = 1 (FID - 1 TOO).
1779
1780 015010 004737 015102 CCB9: JSR PC,CCB30
1781 015014 100137 024613 057024 .WORD 100137,24613,57024,60137 ; OPERAND.
1782 015024 100137 024613 057024 .WORD 100137,24613,57024,60137 ; RESULT
1783 015034 044217 .WORD 44217 ; FPS (FID = FIUV - 1).
1784 015036 144214 000014 .WORD 144214,14 ; FPS, FEC AFTER.
1785
1786 015042 000137 015364 JMP CCBDONE
1787
1788 ; SUBROUTINE TO SET-UP, EXECUTE, AND CHECK RESULTS OF
1789 ; 'NEG', 'ABSD', AND 'TSTD' INSTRUCTIONS.
1790
1791 ; CALL:
1792 ; JSR PC,CCBXX ; 10, 20, OR 30.
1793 ; .WORD X,X,X,X ; OPERAND.
1794 ; .WORD X,X,X,X ; EXPECTED RESULT.
1795 ; .WORD X ; FPS BEFORE EXECUTION.
1796 ; .WORD X,X ; EXPECT FPS, FEC AFTER.
1797 ; ; RETURN TO CALL+26
1798
1799 ; THE PC OF THE CALLING SEQUENCE IS SAVED AS 'ERROR PC' IN
1800 ; CASE OF ERROR. THE APPROPRIATE TEST INSTRUCTION AND ERROR
1801 ; MESSAGE IS SET-UP ACCORDING TO THE ENTRY POINT. IF THE
1802 ; EXPECTED FEC IS NON-ZERO, THE ACTUAL FEC WILL BE TESTED,
1803 ; OTHERWISE, THE FEC IS INVALID (TREATED AS 0).
1804
1805 015046 012701 170710 CCB10: MOV #NEG+10,R1 ; NEG OPCODE...
1806 015052 012702 042516 MOV #'NE,R2 ; ...AND ERROR TEXT.
1807 015056 012703 042107 MOV #'GD,R3
1808 015062 000406 SKP2+2+2
1809 015064 012701 170610 CCB20: MOV #ABSD+10,R1 ; ABSD OPCODE...
1810 015070 012702 041101 MOV #'AB,R2 ; ...AND ERROR TEXT.
1811 015074 012703 042123 MOV #'SD,R3
1812 015100 000406 SKP2+2+2
1813 015102 012701 170510 CCB30: MOV #TSTD+10,R1 ; TSTD OPCODE...
1814 015106 012702 051524 MOV #'TS,R2 ; ...AND ERROR TEXT.
1815 015112 012703 042124 MOV #'TD,R3
1816
1817 015116 010137 015254 1$: MOV R1,3$ ; SET CURRENT OPCODE...

```

1818	015122	010237	040044		MOV	R2,EM76X		;...AND ERROR MESSAGE TEXT.
1819	015126	010337	040046		MOV	R3,EM76X+2		
1820	015132	012637	001164		MOV	(SP)+,\$TMP2		; SAVE CALL PC AS 'ERROR PC'.
1821	015136	104411			LUPERR			:: LOOP HERE ON ERROR IF SWR9 1.
1822	015140	013701	001164		MOV	\$TMP2,R1		; ARG POINTER => R1.
1823	015144	010137	001166		MOV	R1,\$TMP3		; POINTS TO ORIGINAL OPERAND.
1824	015150	062701	000010		ADD	#10,R1		
1825	015154	010137	001170		MOV	R1,\$TMP4		; POINTS TO EXPECTED RESULT.
1826	015160	062701	000012		ADD	#12,R1		
1827	015164	010137	001174		MOV	R1,\$TMP6		; POINTS TO EXPECTED STATUS.
1828	015170	012737	015350	001172	MOV	#20,\$TMP5		; POINTS TO FINAL DATA.
1829	015176	012737	015360	001176	MOV	#21,\$TMP7		; POINTS TO FINAL STATUS.
1830								
1831	015204	013700	001164		MOV	\$TMP2,R0		
1832	015210	012701	015350		MOV	#20\$,R1		
1833	015214	012702	000004		MOV	#4,R2		
1834	015220	012021		2\$:	MOV	(R0)+,(R1)+		; COPY OPERAND TO BUFFER 20\$.
1835	015222	077202			SOB	R2,2\$		
1836	015224	005037	015360		CLR	21\$; AND CLEAR STATUS BUFFER.
1837	015230	005037	015362		CLR	21\$+2		
1838								
1839	015234	013701	001164		MOV	\$TMP2,R1		
1840	015240	016100	000020		MOV	20(R1),R0		; GET INITIAL FPS...
1841	015244	170100			LDFPS	R0		;...AND SET IT.
1842	015246	012700	015350		MOV	#20\$,R0		; SET FDST.
1843	015252	000241			241			
1844	015254	170710		3\$:	NEG	(R0)		; NEG, OR ABSD, OR TSTD.
1845	015256	000241			241			
1846	015260	170200			STFPS	R0		
1847	015262	010037	015360		MOV	R0,21\$; GET FPS.
1848	015266	005761	000024		TST	24(R1)		; IF EXPECTED FEC NON-ZERO...
1849	015272	001403			BEQ	4\$		
1850	015274	170300			STST	R0		;... GET IT TOO.
1851	015276	010037	015362		MOV	R0,21\$+2		
1852	015302	013737	001170	015314	4\$:	MOV	\$TMP4,5\$; SET EXPECTED DATA POINTER.
1853	015310	004537	004062		JSR	R5,CHECK4		; AND CHECK RESULTING DATA.
1854	015314	000000	015350		5\$:	0,20\$		
1855	015320	001010			BNE	7\$; BR IF WRONG.
1856	015322	013737	001174	015334	MOV	\$TMP6,6\$; SET EXPECTED STATUS POINTER.
1857	015330	004537	004052		JSR	R5,CHECK2		; AND CHECK STATUS.
1858	015334	000000	015360		6\$:	0,21\$		
1859	015340	001401			BEQ	8\$; BR IF EVERYTHING OK.
1860								
1861	015342	104076		7\$:	ERROR	76		; RESULT OR STATUS WRONG.
1862								
1863	015344	000161	000026	8\$:	JMP	26(R1)		; RETURN.
1864								
1865	015350	000000	000000	000000	20\$:	.WORD	0,0,0,0	; WORKING DATA BUFFER.
1866	015360	000000	000000		21\$:	.WORD	0,0	; FINAL STATUS BUFFER.
1867								
1868	015364				CCBDONE:			
(1)	015364	104412			CLRFPS			:: CLEAR FP STATUS...
(3)	015366	000400			BR	TST33		::...AND PROCEED.
1869								

```
1876 .....  
(3) *TEST 33 LDFPS -- SRC MODE 1  
(4) *  
(4) * TEST FSRC MODE 1, USING THE LDFPS INSTRUCTION.  
(4) *  
(3) .....  
(2) 015370 000004 TST33: SCOPE  
1877  
1878 015372 DDB1:  
(1) 015372 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
1879 015374 012737 015504 000004 MOV #DDB04,ERRVEC ;: IN CASE IT TRAPS.  
1880 015402 012737 015432 001164 MOV #1$, $TMP2  
1881 015410 012700 015520 MOV #DDB10,R0 ;: SET FSRC.  
1882 015414 010037 001166 MOV R0,$TMP3 ;: SAVE AS R0 BEFORE.  
1883 015420 012701 147517 MOV #147517,R1 ;: FPS WORD TO TEST.  
1884 015424 010110 MOV R1,(R0) ;: PUT DATA IN BUFFER.  
1885 015426 010137 001170 MOV R1,$TMP4 ;: AND SAVE AS EXPECTED FPS.  
1886 015432 170110 1$: LDFPS (R0) ;: TEST INSTRUCTION.  
1887 015434 000241 241  
1888 015436 020037 001166 CMP R0,$TMP3 ;: R0 SHOULD BE UNCHANGED.  
1889 015442 001407 BEQ 2$ ;: BR IF SO.  
1890 015444 013737 001166 001170 MOV $TMP3,$TMP4  
1891 015452 010037 001172 MOV R0,$TMP5  
1892 015456 104100 ERROR 100 ;: R0 BAD.  
1893 015460 000410 BR 3$  
1894  
1895 015462 170200 2$: STFPS R0  
1896 015464 010037 001166 MOV R0,$TMP3 ;: GET FPS.  
1897 015470 023737 001166 001170 CMP $TMP3,$TMP4 ;: CHECK IT.  
1898 015476 001401 BEQ 3$ ;: BR IF RIGHT.  
1899 015500 104101 ERROR 101 ;: IT'S NOT.  
1900 015502 000410 3$: BR DDBDONE  
1901  
1902 ;: TEST INSTR TRAPPED.  
1903  
1904 015504 011637 001164 DDB04: MOV (SP),$TMP2  
1905 015510 022626 CMP (SP)+,(SP)+  
1906 015512 104102 ERROR 102  
1907 015514 000403 BR DDBDONE  
1908  
1909 015516 177777 DDB10: -1  
1910 015520 147517 ;: TEST FPS.  
1911 015522 177777 -1  
1912  
1913 015524 DDBDONE:  
(1) 015524 104412 CLRFP ;: CLEAR FP STATUS...  
(3) 015526 000400 BR TST34 ;: ...AND PROCEED.
```

```

1915 .....
(3) *TEST 34 LDFPS -- SRC MODE 2
(4) *
(4) * TEST FSRC MODE 2, USING THE LDFPS INSTRUCTION.
(4) *
(3) .....
(2) 015530 000004 TST34: SCOPE
1916
1917 015532 EEB1:
(1) 015532 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.
1918 015534 012737 015646 000004 MOV #EEB04,ERRVEC
1919 015542 012737 015574 001164 MOV #1$,STMP2
1920 015550 012700 015662 MOV #EEB10,R0 ; SET FSRC.
1921 015554 010037 001166 MOV R0,$TMP3 ; SAVE AS R0 BEFORE.
1922 015560 012701 145212 MOV #145212,R1 ; SET TEST FPS WORD.
1923 015564 010137 001170 MOV R1,$TMP4
1924 015570 010137 015662 MOV R1,EEB10
1925 015574 170120 1$: LDFPS (R0)+ ;TEST INSTRUCTION.
1926 015576 000241 241
1927 015600 020027 015664 CMP R0,#EEB10+2 ; AUTO-INCR OK ??
1928 015604 001407 BEQ 2$ ; BR IF SO.
1929 015606 012737 015664 001170 MOV #EEB10+2,$TMP4
1930 015614 010037 001172 MOV R0,$TMP5
1931 015620 104103 ERROR 103 ; RO BAD.
1932 015622 000410 BR 3$
1933
1934 015624 170200 2$: STFPS R0 ; GET FPS.
1935 015626 010037 001166 MOV R0,$TMP3
1936 015632 023737 001166 001170 CMP $TMP3,$TMP4 ; FPS RIGHT ??
1937 015640 001401 BEQ 3$ ; BR IF SO.
1938 015642 104104 ERROR 104 ; NO.
1939 015644 000410 3$: BR EEBDONE
1940
1941 ; TEST INSTR TRAPPED.
1942
1943 015646 011637 001164 EEB04: MOV (SP),$TMP2
1944 015652 022626 CMP (SP)+,(SP)+
1945 015654 104105 ERROR 105
1946 015656 000403 BR EEBDONE
1947
1948 015660 177777 EEB10: -1
1949 015662 145212 ; TEST FPS.
1950 015664 177777 -1
1951
1952 015666 EEBDONE:
(1) 015666 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 015670 000400 BR TST35 ;:...AND PROCEED.
    
```

```

1954 .....
(3) *TEST 35          LDFPS -- SRC MODE 4
(4) *
(4) * TEST FSRC MODE 4, USING THE LDFPS INSTRUCTION.
(4) *
(3) .....
(2) 015672 000004 TST35: SCOPE
1955
1956 015674 FFB1:
(1) 015674 104411      LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
1957 015676 012737 016010 000004      MOV #FFB04,ERRVEC
1958 015704 012737 015736 001164      MOV #1$, $TMP2
1959 015712 012700 016034          MOV #FFB10+2,R0 ; SET FSRC
1960 015716 010037 001166          MOV R0,$TMP3    ; SAVE AS R0 BEFORE.
1961 015722 012701 105252          MOV #105252,R1  ; TEST FPS WORD.
1962 015726 010137 001170          MOV R1,$TMP4    ; SAVE AS EXPECTED.
1963 015732 010160 177776          MOV R1,-2(R0)  ; SET IN BUFFER.
1964 015736 170140          1$: LDFPS -(R0)  ; TEST.
1965 015740 000241          241
1966 015742 020027 016032          CMP R0,#FFB10  ; AUTO-DECR OK ??
1967 015746 001407          BEQ 2$         ; YUP.
1968 015750 012737 016032 001170      MOV #FFB10,$TMP4 ; NOPE.
1969 015756 010037 001172          MOV R0,$TMP5
1970 015762 104106          ERROR 106     ; R0 BAD.
1971 015764 000410          BR 3$
1972
1973 015766 170200          2$: STFPS R0    ; GET FPS.
1974 015770 010037 001166          MOV R0,$TMP3
1975 015774 023737 001166 001170      CMP $TMP3,$TMP4 ; FPS RIGHT ??
1976 016002 001401          BEQ 3$         ; YUP.
1977 016004 104107          ERROR 107     ; NOPE.
1978 016006 000416          3$: BR FFBDONE
1979
1980      ; TEST INSTR TRAPPED.
1981
1982 016010 011637 001164      FFB04: MOV (SP),$TMP2
1983 016014 022626          CMP (SP)+,(SP)+
1984 016016 104110          ERROR 110
1985 016020 000411          BR FFBDONE
1986
1987 016022 177777 177777 177777      FFB10: -1,-1,-1,-1
1988 016032 105252          105252 ; TEST FPS WORD.
1989 016034 177777 177777 177777      -1,-1,-1,-1
1990
1991 016044          FFBDONE:
(1) 016044 104412          CLRFPS        ;; CLEAR FP STATUS...
(3) 016046 000400          BR TST36     ;; ...AND PROCEED.

```

```
1993
(3)
(4)
(4)
(4)
(3)
(2) 016050 000004
1994
1995 016052
(1) 016052 104411
1996 016054 012737 016166 000004
1997 016062 012737 016114 001164
1998 016070 012700 016216
1999 016074 010037 001166
2000 016100 012701 103456
2001 016104 010137 001170
2002 016110 010137 016206
2003 016114 170130
2004 016116 000241
2005 016120 020027 016220
2006 016124 001407
2007 016126 012737 016220 001170
2008 016134 010037 001172
2009 016140 104111
2010 016142 000410
2011
2012 016144 170200
2013 016146 010037 001166
2014 016152 023737 001166 001170
2015 016160 001401
2016 016162 104112
2017 016164 000420
2018
2019
2020
2021 016166 011637 001164
2022 016172 022626
2023 016174 104113
2024 016176 000413
2025
2026 016200 177777 177777 177777
2027 016206 103456
2028 016210 177777 177777 177777
2029 016216 016206
2030 016220 177777 177777 177777
2031
2032 016226
(1) 016226 104412
(3) 016230 000400
```

```
*****
*TEST 36 LDFPS -- SRC MODE 3
*
* TEST FSRC MODE 3, USING THE LDFPS INSTRUCTION.
*
*****
ST36: SCOPE

GGB1:
LUPERR ;; LOOP HERE ON ERROR IF SWR9 1.
MOV #GGB04,ERRVEC
MOV #1$, $TMP2
MOV #GGB11,R0 ; SET FSRC.
MOV R0,$TMP3 ; SAVE IT
MOV #103456,R1 ; TEST FPS WORD.
MOV R1,$TMP4 ; SAVE IT.
MOV R1,GGB10 ; HERE TOO.
1$: LDFPS @ (R0)+ ;TEST INSTRUCTION.
241
CMP R0,#GGB11+2 ; AUTO-INCR OK ??
BEQ 2$
MOV #GGB11+2,$TMP4
MOV R0,$TMP5
ERROR 111 ; R0 BAD.
BR 3$

2$: STFPS R0
MOV R0,$TMP3
CMP $TMP3,$TMP4 ; FPS RIGHT ??
BEQ 3$
ERROR 112 ; FPS WRONG.
3$: BR GGBDONE

; TEST INSTR TRAPPED.

GGB04: MOV (SP),$TMP2
CMP (SP)+,(SP)+
ERROR 113
BR GGBDONE

GGB10: -1,-1,-1 ; TEST FPS WORD.
103456
GGB11: GGB10 ; INDIRECT POINTER.
-1,-1,-1

GGBDONE:
CLRFP
BR TST37 ;; CLEAR FP STATUS...
;;...AND PROCEED.
```



```

2034 .....
(3) *TEST 37 LDFPS -- SRC MODE 5
(4) *
(4) * TEST FSRC MODE 5, USING THE LDFPS INSTRUCTION.
(4) *
(3) .....
(2) 016232 000004 TST37: SCOPE
2035
2036 016234 HHB1:
(1) 016234 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
2037 016236 012737 016350 000004 MOV #HHB04,ERRVEC
2038 016244 012737 016276 001164 MOV #1$, $TMP2
2039 016252 012700 016402 MOV #HHB11+2,R0 ; SET FSRC
2040 016256 010037 001166 MOV R0,$TMP3 ; SAVE IT.
2041 016262 012701 045412 MOV #45412,R1 ; TEST FPS WORD.
2042 016266 010137 001170 MOV R1,$TMP4 ; SAVE IT.
2043 016272 010137 016370 MOV R1,HHB10 ; HERE TOO.
2044 016276 170150 1$: LDFPS @-(R0) ;TEST INSTRUCTION.
2045 016300 000241 241
2046 016302 020027 016400 CMP R0,#HHB11 ; AUTO-DECR OK ??
2047 016306 001407 BEQ 2$
2048 016310 012737 016400 001170 MOV #HHB11,$TMP4 ; NO
2049 016316 010037 001172 MOV R0,$TMP5
2050 016322 104114 ERROR 114 ; RO BAD.
2051 016324 000410 BR 3$
2052
2053 016326 170200 2$: STFPS R0 ; GET FPS.
2054 016330 010037 001166 MOV R0,$TMP3
2055 016334 023737 001166 001170 CMP $TMP3,$TMP4 ; FPS RIGHT ??
2056 016342 001401 BEQ 3$
2057 016344 104115 ERROR 115 ; NO.
2058 016346 000420 BR HHBDONE
2059
2060 ; TEST INSTR TRAPPED.
2061
2062 016350 011637 001164 HHB04: MOV (SP),$TMP2
2063 016354 022626 CMP (SP)+,(SP)+
2064 016356 104116 ERROR 116
2065 016360 000413 BR HHBDONE
2066
2067 016362 177777 177777 177777 HHB10: -1,-1,-1
2068 016370 045412 HHB10: 45412 ; TEST FPS WORD.
2069 016372 177777 177777 177777 HHB11: -1,-1,-1
2070 016400 016370 HHB11: HHB10 ; POINTER.
2071 016402 177777 177777 177777 HHB11: -1,-1,-1
2072
2073 016410 HHBDONE:
(1) 016410 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 016412 000400 BR TST40 ;:...AND PROCEED.
  
```

```
2075 .....  
(3) *TEST 40 LDFPS -- SRC MODE 6  
(4) *  
(4) * TEST FSRC MODE 6, USING THE LDFPS INSTRUCTION.  
(4) *  
(3) .....  
(2) 016414 000004 TST40: SCOPE  
2076  
2077 016416 JJB1: LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.  
(1) 016416 104411 MOV #JJB04,ERRVEC ;: LOOP HERE ON ERROR IF SWR9 1.  
2078 016420 012737 016534 000004 MOV #1$, $TMP2  
2079 016426 012737 016460 001164 MOV #JJB10-241,RO ; SET FSRC.  
2080 016434 012700 016307 MOV R0, $TMP3 ; SAVE IT.  
2081 016440 010037 001166 MOV #46543,R1 ; TEST FPS WORD.  
2082 016444 012701 046543 MOV R1, $TMP4 ; SAVE IT.  
2083 016450 010137 001170 MOV R1, JJB10 ; HERE TOO.  
2084 016454 010137 016550 1$: LDFPS 241(RO) ; TEST INSTRUCTION.  
2085 016460 170160 000241 241  
2086 016464 000241 CMP R0, $TMP3 ; RO UNCHANGED ??  
2087 016466 020037 001'66 BEQ 2$  
2088 016472 001407 MOV $TMP3, $TMP4 ; NO.  
2089 016474 013737 001166 001170 MOV R0, $TMP5  
2090 016502 010037 001172 ERROR 117 ; RO BAD.  
2091 016506 104117 BR 3$  
2092 016510 000410  
2093  
2094 016512 170200 2$: STFPS R0 ; GET FPS.  
2095 016514 010037 001166 MOV R0, $TMP3  
2096 016520 023737 001166 001170 CMP $TMP3, $TMP4 ; FPS RIGHT ??  
2097 016526 001401 BEQ 3$  
2098 016530 104120 ERROR 120 ; NO.  
2099 016532 000410 3$: BR JJB DONE  
2100  
2101 ;: TEST INSTR TRAPPED.  
2102  
2103 016534 011637 001164 JJB04: MOV (SP), $TMP2  
2104 016540 022626 CMP (SP)+, (SP)+  
2105 016542 104121 ERROR 121  
2106 016544 000403 BR JJB DONE  
2107  
2108 016546 177777 -1  
2109 016550 046543 JJB10: 46543 ; TEST FPS WORD.  
2110 016552 177777 -1  
2111  
2112 016554 JJB DONE:  
(1) 016554 104412 CLR FPS ;: CLEAR FP STATUS...  
(3) 016556 000400 BR TST41 ;: ...AND PROCEED.
```

```
2114 .....  
(3) *TEST 41 LDFPS -- SRC MODE 7  
(4) *  
(4) * TEST FSRC MODE 7, USING THE LDFPS INSTRUCTION.  
(4) *  
(3) .....  
(2) 016560 000004 TST41: SCOPE  
2115  
2116 016562 KKB1:  
(1) 016562 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
2117 016564 012737 016700 000004 MOV #KKB04,ERRVEC  
2118 016572 012737 016624 001164 MOV #1$, $TMP2  
2119 016600 012700 016467 MOV #KKB11-241,R0 ; SET FSRC  
2120 016604 010037 001166 MOV R0,$TMP3 ; SAVE IT.  
2121 016610 012701 004547 MOV #4547,R1 ; TEST FPS WORD.  
2122 016614 010137 001170 MOV R1,$TMP4 ; SAVE IT.  
2123 016620 010137 016720 MOV R1,KKB10 ; HERE TOO  
2124 016624 170170 000241 1$: LDFPS @241(R0) ;TEST INSTRUCTION.  
2125 016630 000241 241  
2126 016632 020037 001166 CMP R0,$TMP3 ; R0 UNCHANGED ??  
2127 016636 001407 BEQ 2$  
2128 016640 013737 001166 001170 MOV $TMP3,$TMP4  
2129 016646 010037 001172 MOV R0,$TMP5  
2130 016652 104122 ERROR 122 ; R0 BAD.  
2131 016654 000410 BR 3$  
2132  
2133 016656 170200 2$: STFPS R0  
2134 016660 010037 001166 MOV R0,$TMP3  
2135 016664 023737 001166 001170 CMP $TMP3,$TMP4  
2136 016672 001401 BEQ 3$  
2137 016674 104123 ERROR 123 ; FPS BAD.  
2138 016676 000420 3$: BR KKBDONE  
2139  
2140 ;: TEST INSTR TRAPPED.  
2141  
2142 016700 011637 001164 KKB04: MOV (SP),$TMP2  
2143 016704 022626 CMP (SP)+,(SP)+  
2144 016706 104124 ERROR 124  
2145 016710 000413 BR KKBDONE  
2146  
2147 016712 177777 177777 177777 KKB10: -1,-1,-1  
2148 016720 004547 004547 ; TEST FPS WORD.  
2149 016722 177777 177777 177777 KKB11: -1,-1,-1  
2150 016730 016720 KKB11: KKB10 ; POINTER.  
2151 016732 177777 177777 177777 -1,-1,-1  
2152  
2153 KKBDONE:  
(1) 016740 104412 CLRFP  
(3) 016742 000400 BR TST42 ;: CLEAR FP STATUS...  
 ;: ...AND PROCEED.
```

2155
(3)
(4)
(4)
(4)
(3)
(2) 016744 000004
2156
2157
2158
2159
2160
2161 016746
(1) 016746 104411
2162 016750 012737 017052 000004
2163 016756 012700 005741
2164 016762 012701 017016
2165 016766 010021
2166 016770 010021
2167 016772 010021
2168 016774 010021
2169 016776 170011
2170 017000 170012
2171 017002 012737 017014 001164
2172 017010 012701 017026
2173 017014 177027
2174 017016 005741
2175 017020 005741
2176 017022 005741
2177 017024 005741
2178 017026 020127 017020
2179 017032 001406
2180
2181 017034 010137 001166
2182 017040 012737 017020 001170
2183 017046 104125
2184
2185 017050 000404
2186
2187
2188
2189 017052 011637 001164
2190 017056 022626
2191 017060 104126
2192
2193 017062
(1) 017062 104412
(3) 017064 000400

```
*****  
: *TEST 42 LDCLD -- SRC MODE 2 WITH GR7  
: *  
: * TEST FSRC MODE 2(IMMEDIATE), USING THE LDCLD INSTRUCTION.  
: *  
*****  
TST42: SCOPE  
: ALL WE CARE ABOUT IS THE RETURN PC ON THIS ONE.  
: WE'LL SET THE 'FL' BIT TO TRY AND BREAK IT.  
: 'LDCLD' ITSELF IS TESTED LATER (ALONG WITH THE OTHERS).  
: LLB1:  
LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
MOV #LLB04,ERRVEC ; IN CASE IT TRAPS.  
MOV #5741,R0  
MOV #1$+2,R1  
MOV R0,(R1)+ ; IN CASE WE'VE SCREWED UP...  
MOV R0,(R1)+ ;...OUR IN-LINE CODE.  
MOV R0,(R1)+  
MOV R0,(R1)+  
SETD ; FD...  
SETL ;...AND FL = 1.  
MOV #1$,STMP2  
MOV #1$+12,R1 ; TO CALCULATE FINAL PC.  
1$: LDCLD (PC)+,AC0 ;TEST INSTRUCTION.  
5741  
5741  
5741  
5741  
5741  
CMP R1,#1$+4 ; RETURN TO RIGHT PC ??  
BEQ 2$ ; YUP, THAT'S ALL THERE IS.  
MOV R1,STMP3  
MOV #1$+4,STMP4  
ERROR 125 ; FINAL PC WRONG.  
2$: BR LLBDONE  
: TEST INSTRUCTION TRAPPED TO 4.  
: LLB04:  
MOV (SP),STMP2  
CMP (SP)+,(SP)+  
ERROR 126  
LLBDONE:  
CLRFP ;: CLEAR FP STATUS...  
BR TST43 ;...AND PROCEED.
```

```

2195 .....
(3) *TEST 43 LDCLD -- SRC MODE 2
(4) *
(4) * TEST FSRC MODE 2(AUTO-INCR), USING THE LDCLD INSTRUCTION.
(4) *
(3) .....
(2) 017066 000004 TST43: SCOPE
2196 .....
2197 .....
2198 .....
2199 017070 MMB1:
(1) 017070 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.
2200 017072 012737 017110 001164 MOV #1$, $TMP2
2201 017100 170011 SETD
2202 017102 170012 SETL
2203 017104 012700 017146 MOV #10$, R0 ; SET FSRC.
2204 017110 177020 1$: LDCLD (R0)+, ACO ; TEST INSTRUCTION.
2205 017112 000241 241
2206 017114 020027 017152 CMP R0, #10$+4 ; AUTO-INCR OK ??
2207 017120 001411 BEQ 2$ ; YUP. THAT'S IT !.
2208
2209 017122 012737 017146 001166 MOV #10$, $TMP3 ; R0 BEFORE.
2210 017130 012737 017152 001170 MOV #10$+4, $TMP4 ; EXPECTED.
2211 017136 010037 001172 MOV R0, $TMP5 ; READ.
2212 017142 104127 ERROR 127
2213
2214 017144 000404 2$: BR MMBDONE
2215
2216 017146 000000 000001 000002 10$: .WORD 0,1,2,3 ; DON'T CARE ABOUT DATA.
2217
2218 017156 MMBDONE:
(1) 017156 104412 CLRFP
(3) 017160 000400 BR TST44 ;; CLEAR FP STATUS...
;; ...AND PROCEED.
    
```

2226
(3)
(4)
(4)
(4)
(4)
(3)
(2) 017162 000004
2227
2228
2229
2230 017164 004737 017666
2231 017170 000000 000000
2232 017174 000000 000000
2233 017200 000017 000004
2234
2235
2236
2237 017204 004737 017666
2238 017210 000000 177777
2239 017214 000000 000000
2240 017220 000017 000004
2241
2242
2243
2244 017224 004737 017666
2245 017230 000000 000000
2246 017234 000000 000000
2247 017240 000117 000104
2248
2249
2250
2251 017244 004737 017666
2252 017250 040000 000000
2253 017254 043600 000000
2254 017260 000017 000000
2255
2256
2257
2258 017264 004737 017666
2259 017270 000001 000000
2260 017274 040200 000000
2261 017300 000017 000000
2262
2263
2264
2265 017304 004737 017666
2266 017310 000252 000000
2267 017314 042052 000000
2268 017320 000017 000000
2269
2270
2271
2272 017324 004737 017666
2273 017330 140000 000000
2274 017334 143600 000000

```
*****
*TEST 44      LDCIF AND LDCLF -- SRC MODE 1
*
* TEST THE LDCIF/LDCLF INSTRUCTION, USING VARIOUS OPERANDS.
* CONVERT FROM 'I' OR 'L' TO 'F' (SINGLE-PRECISION FLOATING).
*****
TST44: SCOPE
:
: INTEGER = 0, FL = 0.
:
NNB1: JSR      PC,NNBSUB
      .WORD    0,0           ;FSRC OPERAND.
      .WORD    0,0           ;EXPECTED RESULT.
      .WORD    17,4          ;FPS BEFORE AND AFTER.
:
: INTEGER = 0, FL = 0.
:
NNB2: JSR      PC,NNBSUB
      .WORD    0,-1          ;FSRC OPERAND.
      .WORD    0,0           ;EXPECTED RESULT.
      .WORD    17,4          ;FPS BEFORE AND AFTER.
:
: INTEGER = 0, FL = 1.
:
NNB3: JSR      PC,NNBSUB
      .WORD    0,0           ;FSRC OPERAND.
      .WORD    0,0           ;EXPECTED RESULT.
      .WORD    117,104       ;FPS'S.
:
: INTEGER = POS NON-ZERO, FL = 0.
:
NNB4: JSR      PC,NNBSUB
      .WORD    40000,0        ;FSRC OPERAND.
      .WORD    43600,0        ;EXPECTED RESULT.
      .WORD    17,0          ;FPS'S.
:
: INTEGER = 1, FL = 0.
:
NNB5: JSR      PC,NNBSUB
      .WORD    1,0           ;FSRC OPERAND.
      .WORD    40200,0        ;EXPECTED RESULT.
      .WORD    17,0          ;FPS'S.
:
: INTEGER = POS NON-ZERO, FL = 0.
:
NNB6: JSR      PC,NNBSUB
      .WORD    252,0          ;FSRC OPERAND.
      .WORD    42052,0        ;EXPECTED RESULT.
      .WORD    17,0          ;FPS'S.
:
: INTEGER = NEG NON-ZERO, FL = 0.
:
NNB7: JSR      PC,NNBSUB
      .WORD    -40000,0       ;FSRC OPERAND.
      .WORD    143600,0       ;EXPECTED RESULT.
```

2275	017340	000017	000010	.WORD	17,10	; FPS'S.
2276				:		
2277				:	INTEGER = -1, FL = 0.	
2278				:		
2279	017344	004737	017666	NNB8: JSR	PC,NNBSUB	
2280	017350	177777	000000	.WORD	-1,0	;FSRC OPERAND.
2281	017354	140200	000000	.WORD	140200,0	;EXPECTED RESULT.
2282	017360	000017	000010	.WORD	17,10	; FPS'S.
2283				:		
2284				:	INTEGER = NEG NON-ZERO, FL = 0.	
2285				:		
2286	017364	004737	017666	NNB9: JSR	PC,NNBSUB	
2287	017370	125252	000000	.WORD	125252,0	;FSRC OPERAND.
2288	017374	143652	126000	.WORD	143652,126000	;EXPECTED RESULT.
2289	017400	000017	000010	.WORD	17,10	; FPS'S.
2290				:		
2291				:	INTEGER = POS NON-ZERO, FL = 1.	
2292				:		
2293	017404	004737	017666	NNB10: JSR	PC,NNBSUB	
2294	017410	040000	000000	.WORD	40000,0	;FSRC OPERAND.
2295	017414	047600	000000	.WORD	47600,0	;EXPECTED RESULT.
2296	017420	000117	000100	.WORD	117,100	; FPS'S.
2297				:		
2298				:	INTEGER = 1, FL = 1.	
2299				:		
2300	017424	004737	017666	NNB11: JSR	PC,NNBSUB	
2301	017430	000000	000001	.WORD	0,1	;FSRC OPERAND.
2302	017434	040200	000000	.WORD	40200,0	;EXPECTED RESULT.
2303	017440	000117	000100	.WORD	117,100	; FPS'S.
2304				:		
2305				:	OPERAND = POS NON-ZERO, FL = 1.	
2306				:		
2307	017444	004737	017666	NNB12: JSR	PC,NNBSUB	
2308	017450	000000	000252	.WORD	0,252	;FSRC OPERAND.
2309	017454	042052	000000	.WORD	42052,0	;EXPECTED RESULT.
2310	017460	000117	000100	.WORD	117,100	; FPS'S.
2311				:		
2312				:	INTEGER = NEG NON-ZERO, FL = 1.	
2313				:		
2314	017464	004737	017666	NNB13: JSR	PC,NNBSUB	
2315	017470	140000	000000	.WORD	-40000,0	;FSRC OPERAND.
2316	017474	147600	000000	.WORD	147600,0	;EXPECTED RESULT.
2317	017500	000117	000110	.WORD	117,110	; FPS'S.
2318				:		
2319				:	INTEGER = -1, FL = 1.	
2320				:		
2321	017504	004737	017666	NNB14: JSR	PC,NNBSUB	
2322	017510	177777	177777	.WORD	-1,-1	;FSRC OPERAND.
2323	017514	140200	000000	.WORD	140200,0	;EXPECTED RESULT.
2324	017520	000117	000110	.WORD	117,110	; FPS'S.
2325				:		
2326				:	INTEGER = NEG NON-ZERO, FL = 1, ROUND IT.	
2327				:		
2328	017524	004737	017666	NNB15: JSR	PC,NNBSUB	
2329	017530	125252	125252	.WORD	125252,125252	;FSRC OPERAND.
2330	017534	147652	125253	.WORD	147652,125253	;EXPECTED RESULT.

```
2331 017540 000117 000110 .WORD 117,110 ; FPS'S.
2332
2333 : INTEGER = POS NON-ZERO, FL = 1, ROUND IT.
2334
2335 017544 004737 017666 NNB16: JSR PC,NNBSUB
2336 017550 077777 177500 .WORD 77777,177500 ;FSRC OPERAND.
2337 017554 047777 177777 .WORD 47777,177777 ;EXPECTED RESULT.
2338 017560 000117 000100 .WORD 117,100 ; FPS'S.
2339
2340 : INTEGER = POS NON-ZERO, FL = 1, ROUND.
2341
2342 017564 004737 017666 NNB17: JSR PC,NNBSUB
2343 017570 040000 000100 .WORD 40000,100 ;FSRC OPERAND.
2344 017574 047600 000001 .WORD 47600,1 ;EXPECTED RESULT.
2345 017600 000117 000100 .WORD 117,100 ; FPS'S.
2346
2347 : INTEGER = POS NON-ZERO, FL = 1, TRUNCATE.
2348
2349 017604 004737 017666 NNB18: JSR PC,NNBSUB
2350 017610 040000 000100 .WORD 40000,100 ;FSRC OPERAND.
2351 017614 047600 000000 .WORD 47600,0 ;EXPECTED RESULT.
2352 017620 000157 000140 .WORD 157,140 ; FPS'S.
2353
2354 : INTEGER = NEG NON-ZERO (MAX NEG), FL = 0.
2355
2356 017624 004737 017666 NNB19: JSR PC,NNBSUB
2357 017630 100000 000000 .WORD 100000,0 ;FSRC OPERAND.
2358 017634 144000 000000 .WORD 144000,0 ;EXPECTED RESULT.
2359 017640 000017 000010 .WORD 17,10 ; FPS'S.
2360
2361 : INTEGER = MAX NEG, FL = 1.
2362
2363 017644 004737 017666 NNB20: JSR PC,NNBSUB
2364 017650 100000 000000 .WORD 100000,0 ;FSRC OPERAND.
2365 017654 150000 000000 .WORD 150000,0 ;EXPECTED RESULT.
2366 017660 000117 000110 .WORD 117,110 ; FPS'S.
2367
2368 017664 000475 BR NNBDONE
2369
2370 : SUBROUTINE TO SET-UP, EXECUTE, AND CHECK RESULTS OF
2371 : LDCIF AND/OR LDCIF INSTRUCTIONS.
2372
2373 : CALL:
2374 : JSR PC,NNBSUB
2375 : .WORD X,X ; SRC INTEGER (I OR L).
2376 : .WORD X,X ; EXPECTED RESULT.
2377 : .WORD X,X ; FPS BEFORE AND AFTER CONVERSION.
2378 : ; RETURN TO CALL+14.
2379
2380 : SAVE THE CALL PC AS 'ERROR PC' IN CASE OF ERROR.
2381 : EXECUTE THE LDC_F INSTRUCTION AND CHECK CONVERTED RESULT
2382 : AND FINAL FPS.
2383
2384 017666 012637 001164 NNBSUB: MOV (SP)+,$TMP2 ; SAVE AS 'ERROR PC'.
2385 017672 104411 LUPERR ; ; LOOP HERE ON ERROR IF SWR9 = 1.
2386 017674 013701 001164 MOV $TMP2,R1 ; GET POINTER.
```


2387	017700	016137	000012	001170		MOV	12(R1), \$TMP4	:	EXPECTED FPS.
2388	017706	010137	001172			MOV	R1, \$TMP5	:	POINTS TO SRC INTEGER.
2389	017712	062701	000004			ADD	#4, R1		
2390	017716	010137	001174			MOV	R1, \$TMP6	:	POINTS TO EXP'D RESULT.
2391	017722	013701	001164			MOV	\$TMP2, R1	:	GET POINTER AGAIN...
2392	017726	016100	000010			MOV	10(R1), R0		
2393	017732	170100				LD FPS	R0	:	...AND SET INITIAL FPS.
2394									
2395	017734	010100				MOV	R1, R0	:	SET FSRC.
2396	017736	177010			1\$:	LDCIF	(R0), ACO	:	TEST INSTRUCTION LDCIF/LDCLF.
2397	017740	000241				241			
2398	017742	170200				ST FPS	R0	:	GET FPS.
2399	017744	010037	001166			MOV	R0, \$TMP3		
2400	017750	170011				SETD		:	DOUBLE MODE...
2401	017752	012700	020050			MOV	#10\$, R0		
(1)	017756	174010				STD	ACO, (R0)	::	STORE ACO
2402									
2403	017760	023737	001166	001170		CMP	\$TMP3, \$TMP4	:	FPS RIGHT ??
2404	017766	001010				BNE	3\$:	NO.
2405	017770	013737	001174	020002		MOV	\$TMP6, 2\$:	YES, SET EXP'D POINTER...
2406	017776	004537	004052			JSR	R5, CHECK2	:	...AND CHECK DATA.
2407	020002	000000	020050		2\$:	0, 10\$			
2408	020006	001416				BEQ	5\$:	BR IF BOTH WERE OK.
2409									
2410	020010	112737	000111	041575	3\$:	MOVB	#'I, EM130X	:	ASSUME 'LDCIF' ERROR TEXT.
2411	020016	032761	000100	000010		BIT	#100, 10(R1)	:	WAS 'FL' BIT SET ??
2412	020024	001403				BEQ	4\$:	SKIP NEXT IF NOT.
2413	020026	112737	000114	041575		MOVB	#'L, EM130X	:	YES, CHANGE TO 'LDCLF'.
2414	020034	012737	020050	001176	4\$:	MOV	#10\$, \$TMP7		
2415	020042	104130				ERROR	130	:	RESULT OR FPS WRONG.
2416									
2417	020044	000161	000014		5\$:	JMP	14(R1)	:	RETURN.
2418									
2419	020050	000000	000000	000000	10\$:	.WORD	0, 0, 0, 0	:	SCRATCH BUFFER.
2420									
2421	020060					NNBDONE:			
(1)	020060	104412				CLR FPS		::	CLEAR FP STATUS...
(3)	020062	000400				BR	TST45	::	...AND PROCEED.

```
2429 .....  
  (3) *TEST 45      LDCID AND LDCLD -- SRC MODE 1  
  (4) *  
  (4) * TEST THE LDCID/LDCLD INSTRUCTION, USING VARIOUS OPERANDS.  
  (4) * CONVERT FROM 'I' OR 'L' TO 'D' (DOUBLE-PRECISION FLOATING).  
  (4) *  
  (3) .....  
  (2) 020064 000004 TST45: SCOPE  
2430 :  
2431 : INTEGER 0, FL = 0.  
2432 :  
2433 020066 004737 020400 PPB1: JSR PC,PPBSUB  
2434 020072 000000 000000 .WORD 0,0 ;FSRC OPERAND.  
2435 020076 000000 000000 000000 .WORD 0,0,0,0 ;EXPECTED RESULT.  
2436 020106 000217 000204 .WORD 217,204 ; FPS BEFORE AND AFTER.  
2437 :  
2438 : INTEGER = 0, FL = 0.  
2439 :  
2440 020112 004737 020400 PPB2: JSR PC,PPBSUB  
2441 020116 000000 177777 .WORD 0,-1 ;FSRC OPERAND.  
2442 020122 000000 000000 000000 .WORD 0,0,0,0 ;EXPECTED RESULT.  
2443 020132 000217 000204 .WORD 217,204 ; FPS'S.  
2444 :  
2445 : INTEGER = 0, FL = 1.  
2446 :  
2447 020136 004737 020400 PPB3: JSR PC,PPBSUB  
2448 020142 000000 000000 .WORD 0,0 ;FSRC OPERAND.  
2449 020146 000000 000000 000000 .WORD 0,0,0,0 ;EXPECTED RESULT.  
2450 020156 000317 000304 .WORD 317,304 ; FPS'S.  
2451 :  
2452 : INTERER - POS NON-ZERO, FL = 0.  
2453 :  
2454 020162 004737 020400 PPB4: JSR PC,PPBSUB  
2455 020166 040000 000000 .WORD 4000,0 ;FSRC OPERAND.  
2456 020172 043600 000000 000000 .WORD 43600,0,0,0 ;EXPECTED RESULT.  
2457 020202 000217 000200 .WORD 217,200 ; FPS'S  
2458 :  
2459 : INTEGER = NEG NON-ZERO, FL = 0.  
2460 :  
2461 020206 004737 020400 PPB5: JSR PC,PPBSUB  
2462 020212 140000 000000 .WORD -4000,0 ;FSRC OPERAND.  
2463 020216 143600 000000 000000 .WORD 143600,0,0,0 ;EXPECTED RESULT.  
2464 020226 000217 000210 .WORD 217,210 ; FPS'S.  
2465 :  
2466 : INTEGER = POS NON-ZERO, FL = 1.  
2467 :  
2468 020232 004737 020400 PPB6: JSR PC,PPBSUB  
2469 020236 040000 000000 .WORD 4000,0 ;FSRC OPERAND.  
2470 020242 047600 000000 000000 .WORD 47600,0,0,0 ;EXPECTED RESULT.  
2471 020252 000317 000300 .WORD 317,300 ; FPS'S.  
2472 :  
2473 : INTEGER = 1, FL = 1.  
2474 :  
2475 020256 004737 020400 PPB7: JSR PC,PPBSUB  
2476 020262 000000 000001 .WORD 0,1 ;FSRC OPERAND.  
2477 020266 040200 000000 000000 .WORD 40200,0,0,0 ;EXPECTED RESULT.
```

```
2478 020276 000317 000300 .WORD 317,300 ; FPS'S.
2479
2480 ; INTEGER = MAX POS, FL = 1.
2481
2482 020302 004737 020400 PPB8: JSR PC,PPBSUB
2483 020306 077777 177777 .WORD 77777,177777 ; FSRC OPERAND.
2484 020312 047777 177777 177000 .WORD 47777,177777,177000,0 ; EXPECTED RESULT.
2485 020322 000317 000300 .WORD 317,300 ; FPS'S.
2486
2487 ; INTEGER = MAX NEG, FL = 1.
2488
2489 020326 004737 020400 PPB9: JSR PC,PPBSUB
2490 020332 100000 000000 .WORD 100000,0 ; FSRC OPERAND.
2491 020336 150000 000000 000000 .WORD 150000,0,0,0 ; EXPECTED RESULT.
2492 020346 000317 000310 .WORD 317,310 ; FPS'S.
2493
2494 ; INTEGER = POS NON-ZERO, FL = 1, TRUNCATE.
2495
2496 020352 004737 020400 PPB10: JSR PC,PPBSUB
2497 020356 012345 067012 .WORD 12345,67012 ; FSRC OPERAND.
2498 020362 047247 025560 050000 .WORD 47247,025560,050000,0 ; EXPECTED RESULT.
2499 020372 000357 000340 .WORD 357,340 ; FPS'S.
2500
2501 020376 000475 BR PPBDONE
2502
2503 ; SUBROUTINE TO SET-UP, EXECUTE, AND CHECK RESULTS OF
2504 ; LDCID AND/OR LDCLD INSTRUCTIONS.
2505
2506 ; CALL:
2507 ; JSR PC,PPBSUB
2508 ; .WORD X,X ; SRC INTEGER (I OR L).
2509 ; .WORD X,X,X,X ; EXPECTED DOUBLE RESULT.
2510 ; .WORD X,X ; FPS BEFORE AND AFTER CONVERSION.
2511 ; ; RETURN TO CALL+20.
2512
2513 ; SAVE THE CALL PC AS 'ERROR PC' IN CASE OF ERROR.
2514 ; EXECUTE THE LDC_D INSTRUCTION AND CHECK CONVERTED RESULT
2515 ; AND FINAL FPS.
2516
2517 020400 012637 001164 PPBSUB: MOV (SP)+,$TMP2 ; SAVE AS 'ERROR PC'.
2518 020404 104411 LUPERR ; ; LOOP HERE ON ERROR IF SWR9 = 1.
2519 020406 013701 001164 MOV $TMP2,R1 ; GET POINTER.
2520 020412 016137 000016 001170 MOV 16(R1),$TMP4 ; EXPECTED FPS.
2521 020420 010137 001172 MOV R1,$TMP5 ; POINTS TO SRC INTEGER.
2522 020424 062701 000004 ADD #4,R1
2523 020430 010137 001174 MOV R1,$TMP6 ; POINTS TO EXP'D RESULT.
2524 020434 013701 001164 MOV $TMP2,R1 ; GET POINTER AGAIN...
2525 020440 016100 000014 MOV 14(R1),R0
2526 020444 170100 LDFPS R0 ; ...AND SET INITIAL FPS.
2527
2528 020446 010100 MOV R1,R0 ; SET FSRC.
2529 020450 177010 1$: LDCID (R0),AC0 ; TEST INSTRUCTION LDCID/LDCLD.
2530 020452 000241 241
2531 020454 170200 STFPS R0 ; GET FPS.
2532 020456 010037 001166 MOV R0,$TMP3
2533 020462 170011 SETD ; DOWL MODE...
```

2534	020464	012700	020562			MOV	#10\$,R0		
(1)	020470	174010				STD	AC0,(R0)	::	STORE AC0
2535									
2536	020472	023737	001166	001170		CMP	\$TMP3,\$TMP4	:	FPS RIGHT ??
2537	020500	001010				BNE	3\$:	NO.
2538	020502	013737	001174	020514		MOV	\$TMP6,2\$:	YES, SET EXP'D POINTER...
2539	020510	004537	004062			JSR	R5,CHECK4	:	...AND CHECK DATA.
2540	020514	000000	020562		2\$:		0,10\$		
2541	020520	001416				BEQ	5\$:	BR IF BOTH WERE OK.
2542									
2543	020522	112737	000111	041653	3\$:	MOVB	#'1,EM131X	:	ASSUME 'LDCID' ERROR TEXT.
2544	020530	032761	000100	000014		BIT	#100,14(R1)	:	WAS 'FL' BIT SET ??
2545	020536	001403				BEQ	4\$:	SKIP NEXT IF NOT.
2546	020540	112737	000114	041653		MOVB	#'L,EM131X	:	YES, CHANGE TO 'LDCLD'.
2547	020546	012737	020562	001176	4\$:	MOV	#10\$, \$TMP7		
2548	020554	104131				ERROR	131	:	RESULT OR FPS WRONG.
2549									
2550	020556	000161	000020		5\$:	JMP	20(R1)	:	RETURN.
2551									
2552	020562	000000	000000	000000	10\$:	.WORD	0,0,0,0	:	SCRATCH BUFFER.
2553									
2554	020572					PPBDONE:			
(1)	020572	104412				CLRFPS		::	CLEAR FP STATUS...
(3)	020574	000400				BR	TST46	::	...AND PROCEED.

2562
(3)
(4)
(4)
(4)
(4)
(3)
(2) 020576 000004
2563
2564
2565
2566 020600 004737 021502
2567 020604 012345 067012 034567
2568 020614 000010
2569 020616 042145 067012 034567
2570 020626 047217
2571 020630 047200 000000
2572
2573
2574
2575 020634 004737 021502
2576 020640 123456 070123 045670
2577 020650 000177
2578 020652 177656 070123 045670
2579 020662 047217
2580 020664 047210 000000
2581
2582
2583
2584 020670 004737 021502
2585 020674 073261 057 45 043323
2586 020704 000056
2587 020706 053461 057645 043323
2588 020716 047200
2589 020720 047200 000000
2590
2591
2592
2593 020724 004737 021502
2594 020730 012223 024252 062720
2595 020740 177627
2596 020742 005623 024252 062720
2597 020752 047200
2598 020754 047200 000000
2599
2600
2601
2602 020760 004737 021502
2603 020764 030131 032334 035363
2604 020774 177600
2605 020776 000131 032334 035363
2606 021006 047217
2607 021010 147204 000012
2608
2609
2610

```
*****
*TEST 46          LDEXP -- SRC MODE 0 AND 1
*
* TEST THE LDEXP INSTRUCTION WITH A VARIETY OF OPERANDS.
* BOTH SRC MODES 0 AND 1 ARE TESTED FOR EACH OPERAND SET.
*
*****
TST46: SCOPE
: EXPONENT = 10 (210).
:
QOB1: JSR PC,QOBSUB ; XCT LDEXP, SRC MODE 0 AND 1.
      .WORD 12345,67012,34567,012345 ;ACO OPERAND.
      .WORD 10 ;NEW EXPONENT.
      .WORD 42145,67012,34567,012345 ;RESULT.
      .WORD 47217 ;FPS BEFORE EXECUTION.
      .WORD 47200,0 ; FPS, FEC AFTER.
:
: EXPONENT = 177 (377).
:
QOB2: JSR PC,QOBSUB ; XCT LDEXP, SRC MODE 0 AND 1.
      .WORD 123456,70123,45670,123456 ;ACO OPERAND.
      .WORD 177 ;NEW EXPONENT.
      .WORD 177656,70123,45670,123456 ;RESULT.
      .WORD 47217 ; FPS BEFORE.
      .WORD 47210,0 ; FPS, FEC AFTER.
:
: EXPONENT = 56 (256).
:
QOB3: JSR PC,QOBSUB
      .WORD 73261,057645,43323,101760 ;ACO OPERAND.
      .WORD 56 ; NEW EXPONENT.
      .WORD 53461,057645,43323,101760 ;RESULT.
      .WORD 47200 ;FPS BEFORE EXECUTION.
      .WORD 47200,0 ; FPS, FEC AFTER.
:
: EXPONENT = -151 (027).
:
QOB4: JSR PC,QOBSUB
      .WORD 12223,24252,62720,21222 ;ACO OPERAND.
      .WORD -151 ; NEW EXPONENT.
      .WORD 5623,24252,62720,21222 ;RESULT.
      .WORD 47200 ;FPS BEFORE EXECUTION.
      .WORD 47200,0 ; FPE, FEC AFTER.
:
: EXPONENT = -200 (0), UNDERFLOW, FIU - 1.
:
QOB5: JSR PC,QOBSUB
      .WORD 30131,32334,35363,73031 ;ACO OPERAND.
      .WORD -200 ;NEW EXPONENT.
      .WORD 00131,32334,35363,73031 ;RESULT.
      .WORD 47217 ; FPS BEFORE.
      .WORD 147204,12 ; FPS, FEC AFTER.
:
: EXPONENT = -200 (0), UNDERFLOW, FIU - 1.
:
```

2611	021014	004737	021502		QOB6:	JSR	PC,QOBSUB		
2612	021020	140414	024344	045464		.WORD	140414,24344,45464,74045		:ACO OPERAND.
2613	021030	177600				.WORD	-200		:NEW EXPONENT.
2614	021032	100014	024344	045464		.WORD	100014,24344,45464,74045		:RESULT -0
2615	021042	047217				.WORD	47217		:FPS
2616	021044	147214	000012			.WORD	147214,12		:FPS, FEC.
2617									
2618									
2619									
2620	021050	004737	021502		QOB7:	JSR	PC,QOBSUB		
2621	021054	051525	035455	005675		.WORD	51525,35455,5675,05152		:ACO OPERAND.
2622	021064	177600				.WORD	-200		:NEW EXPONENT.
2623	021066	000000	000000	000000		.WORD	0,0,0,0		:RESULT.
2624	021076	045217				.WORD	45217		:FPS (FIU = 0).
2625	021100	045204	000012			.WORD	45204,12		:FPS, FEC.
2626									
2627									
2628									
2629	021104	004737	021502		QOB8:	JSR	PC,QOBSUB		
2630	021110	061626	062636	046566		.WORD	61626,62636,46566,67606		:ACO OPERAND.
2631	021120	176173				.WORD	-1605		:NEW EXPONENT.
2632	021122	076626	062636	046566		.WORD	76626,62636,46566,67606		:RESULT.
2633	021132	047217				.WORD	47217		:FPS.
2634	021134	147200	000012			.WORD	147200,12		:FPS, FEC.
2635									
2636									
2637									
2638	021140	004737	021502		QOB9:	JSR	PC,QOBSUB		
2639	021144	071727	037475	076777		.WORD	71727,37475,76777,17273		:ACO OPERAND.
2640	021154	160162				.WORD	-17616		:NEW EXPONENT.
2641	021156	000000	000000	000000		.WORD	0,0,0,0		:RESULT.
2642	021166	045217				.WORD	45217		:FPS (FIU = 0).
2643	021170	045204	000012			.WORD	45204,12		:FPS, FEC.
2644									
2645									
2646									
2647	021174	004737	021502		QOB10:	JSR	PC,QOBSUB		
2648	021200	001020	030405	006070		.WORD	01020,30405,06070,00102		:ACO OPERAND.
2649	021210	175777				.WORD	-2001		:NEW EXPONENT.
2650	021212	037620	030405	006070		.WORD	37620,30405,06070,00102		:RESULT.
2651	021222	047217				.WORD	47217		:FPS.
2652	021224	147200	000012			.WORD	147200,12		:FPS, FEC.
2653									
2654									
2655									
2656	021230	004737	021502		QOB11:	JSR	PC,QOBSUB		
2657	021234	012131	014151	016171		.WORD	12131,14151,16171,10111		:ACO OPERAND.
2658	021244	001006				.WORD	1006		:NEW EXPONENT.
2659	021246	041531	014151	016171		.WORD	41531,14151,16171,10111		:RESULT.
2660	021256	047217				.WORD	47217		:FPS.
2661	021260	147202	000010			.WORD	147202,10		:FPS, FEC.
2662									
2663									
2664									
2665	021264	004737	021502		QOB12:	JSR	PC,QOBSUB		
2666	021270	027262	025242	023222		.WORD	27262,25242,23222,21202		:ACO OPERAND.

: EXPONENT = -200 (0), UNDERFLOW, FIU = 0.

: EXPONENT = -1605 (-1405), UNDERFLOW, FIU = 1.

: EXPONENT = -17616 (-17416), UNDERFLOW, FIU = 0.

: EXPONENT = -2001 (-1601), UNDERFLOW, FIU = 1.

: EXPONENT = 1006 (1206), OVERFLOW, FIV = 1.

: EXPONENT = 16115 (16315), OVERFLOW, FIV = 0.

```
2667 021300 016115 .WORD 16115 ;NEW EXPONENT.
2668 .WORD 63262,25242,23222,21202 ; RESULT.
2669 021302 000000 000000 000000 :
2670 021312 046217 .WORD 0,0,0,0 ; FPS (FIV = 0).
2671 .WORD 46217 ; FPS, FEC.
2672 021314 046206 000010 .WORD 46202,0
2673 .WORD 46206,10
2674 : EXPONENT = 10611 (11011), OVERFLOW, FIV = 1.
2675 :
2676 021320 004737 021502 QQB13: JSR PC,QQBSUB
2677 021324 030313 032333 034353 .WORD 30313,32333,34353,36373 ;ACO OPERAND.
2678 021334 010611 .WORD 10611 ; NEW EXPONENT.
2679 021336 002313 032333 034353 .WORD 2313,32333,34353,36373 ;RESULT.
2680 021346 047217 .WORD 47217 ; FPS.
2681 021350 147202 000010 .WORD 147202,10 ; FPS, FEC.
2682 :
2683 : EXPONENT = 16723 (17123), OVERFLOW, FIV = 0.
2684 :
2685 021354 004737 021502 QQB14: JSR PC,QQBSUB
2686 021360 040414 042434 044454 .WORD 40414,42434,44454,46474 ;ACO OPERAND.
2687 021370 016723 .WORD 16723 ;NEW EXPONENT.
2688 .WORD 24614,42434,44454,46474 ; RESULT.
2689 021372 000000 000000 000000 .WORD 0,0,0,0
2690 021402 046217 .WORD 46217 ; FPS (FIV = 0).
2691 .WORD 46202,0 ; FPS, FEC
2692 021404 046206 000010 .WORD 46206,10
2693 :
2694 : EXPONENT = 254 (454), OVERFLOW, FIV = 1.
2695 :
2696 021410 004737 021502 QQB15: JSR PC,QQBSUB
2697 021414 050515 052535 054555 .WORD 50515,52535,54555,56575 ;ACO OPERAND.
2698 021424 000254 .WORD 254 ; NEW EXPONENT.
2699 021426 013115 052535 054555 .WORD 13115,52535,54555,56575 ;RESULT.
2700 021436 047217 .WORD 47217 ; FPS.
2701 021440 147202 000010 .WORD 147202,10 ; FPS, FEC.
2702 :
2703 : EXPONENT = 313 (513), OVERFLOW, FIV = 0.
2704 :
2705 021444 004737 021502 QQB16: JSR PC,QQBSUB
2706 021450 060616 062636 064656 .WORD 60616,62636,64656,66676 ;ACO OPERAND.
2707 021460 000313 .WORD 313 ; NEW EXPONENT.
2708 .WORD 22616,62636,64656,66676 ; RESULT.
2709 021462 000000 000000 000000 .WORD 0,0,0,0
2710 021472 046217 .WORD 46217 ; FPS (FIV = 0).
2711 .WORD 46202,0 ; FPS, FEC.
2712 021474 046206 000010 .WORD 46206,10
2713 :
2714 021500 000546 BR QQBDONE
2715 :
2716 : SUBROUTINE TO SETUP, EXECUTE AND CHECK RESULTING DATA
2717 : AND STATUS FOR THE LDEXP INSTRUCTION.
2718 :
2719 : CALL:
2720 : JSR PC,QQBSUB
2721 : .WORD X,X,X,X ; AC OPERAND.
2722 : .WORD X ; NEW EXPONENT TO LOAD.
```

```

2723      :      .WORD  X,X,X,X      : EXPECTED RESULT.
2724      :      .WORD  X              : FPS BEFORE EXECUTION.
2725      :      .WORD  X,X          : FPS AND FEC EXPECTED AFTER.
2726      :      :                      : RETURN TO CALL+30.
2727      :
2728      :      SAVE THE CALL PC AS 'ERROR PC' IN CASE OF ERROR.
2729      :
2730      021502 012637 001164      QOBSUB: MOV      (SP)+,$TMP2      ; SAVE CALL AS 'ERROR PC'.
2731      021506 005037 021772      CLR      7$              ; SET FLAG = SRC MODE 0.
2732      021512 112737 000177 041737  MOVB     #177,EM132X
2733      021520 112737 000177 041742  MOVB     #177,EM132X+3      ; SET ERROR TEXT = LDEXP R0,ACO
2734      021526 000406              SKP3+3      ;...AND SKIP NEXT 2.
2735      021530 112737 000050 041737 8$:  MOVB     #'(,EM132X
2736      021536 112737 000051 041742  MOVB     #'),EM132X+3      ; CHANGE TEXT 'R0' TO '(R0)'.
2737      :
2738      021544 104411              LUPERR      ;; LOOP HERE ON ERROR IF SWR9 1.
2739      021546 013701 001164      MOV      $TMP2,R1        ; ARG POINTER => R1.
2740      021552 010137 001166      MOV      R1,$TMP3        ; POINTS TO AC OPERAND.
2741      021556 062701 000010      ADD      #10,R1
2742      021562 010137 001170      MOV      R1,$TMP4        ; POINTS TO SRC OPERAND (EXPON).
2743      021566 062701 000002      ADD      #2,R1
2744      021572 010137 001172      MOV      R1,$TMP5        ; POINTS TO EXP'D RESULT.
2745      021576 062701 000012      ADD      #12,R1
2746      021602 010137 001176      MOV      R1,$TMP7        ; POINTS TO EXP'D STATUS.
2747      :
2748      021606 013701 001164      MOV      $TMP2,R1        ; GET ARG POINTER AGAIN.
2749      021612 016100 000022      MOV      22(R1),R0
2750      021616 170100              LDFPS     R0              ; SET INITIAL FPS.
2751      021620 013700 001166      MOV      $TMP3,R0        ; SET UP ACO.
2752      021624 172410              LDD      (R0),ACO
2753      021626 013700 001170      MOV      $TMP4,R0        ; SET FSRC...
2754      021632 011037 001170      MOV      (R0),$TMP4      ;...SAVE EXPON FOR ERROR OUTPUT.
2755      021636 005737 021772      TST      7$              ; MODE 0 ??
2756      021642 100403              BMI      1$              ; BR IF NOT.
2757      021644 011000              MOV      (R0),R0        ; SET FOR SRC MODE 0.
2758      021646 176400              LDEXP    R0,ACO         ; XCT LDEXP, MODE 0.
2759      021650 000401              SKP1
2760      021652 176410              1$:  LDEXP    (R0),ACO      ; XCT LDEXP, MODE 1.
2761      021654 000241              241
2762      021656 170200              STFPS    R0
2763      021660 010037 022012      MOV      R0,11$         ; GET FPS.
2764      021664 005000              CLR      R0
2765      021666 005761 000026      TST      26(R1)         ; IF EXP'D FEC IS NON-ZERO...
2766      021672 001401              BEQ      2$
2767      021674 170300              STST     R0              ;...GET FEC TOO.
2768      021676 010037 022014      2$:  MOV      R0,11$+2
2769      021702 170011              SETD
2770      021704 012700 022002      MOV      #10$,R0        ; INSURE DOUBLE MODE STILL SET.
2771      (1) 021710 174010              STD      ACO,(R0)       ;; STORE ACO
2772      021712 013737 001172 021724      MOV      $TMP5,3$       ; SET DATA POINTER...
2773      021720 004537 004062              JSR     R5,CHECK4      ;...AND CHECK FINAL DATA.
2774      021724 000000 022002      3$:  0,10$
2775      021730 001010              BNE     5$              ; BR IF WRONG.
2776      021732 013737 001176 021744      MOV      $TMP7,4$       ; SET STATUS POINTER...
2777      021740 004537 004052              JSR     R5,CHECK2      ;...AND CHECK STATUS TOO.
  
```


2778	021744	000000	022012	4\$:	0,11\$		
2779	021750	001407			BEQ	6\$; BR IF BOTH WERE OK.
2780							
2781	021752	012737	022002	5\$:	MOV	#10\$,STMP6	
2782	021760	012737	022012		MOV	#11\$,STMP10	
2783	021766	104132			ERROR	132	; RESULT OR STATUS INCORRECT.
2784							
2785	021770	005127		6\$:	COM	(PC)+	; 1ST PASS ??
2786	021772	000000		7\$:	0		
2787	021774	100655			BMI	8\$; YES, DO IT AGAIN IN SRC MODE 1.
2788	021776	000161	000030		JMP	30(R1)	; AND THEN RETURN.
2789							
2790	022002	000000	000000	10\$:	.WORD	0,0,0,0	; FINAL DATA BUFFER.
2791	022012	000000	000000	11\$:	.WORD	0,0	; FINAL FPS AND FEC.
2792							
2793	022016			QQBDONE:			
(1)	022016	104412			CLRFP		:: CLEAR FP STATUS...
(3)	022020	000400			BR	TST47	::...AND PROCEED.

```

2800 .....
(3) *TEST 47 STFPS -- DST MODE 1
(4) .....
(4) * TEST FDST MODE 1, USING THE STFPS INSTRUCTION.
(4) .....
(3) .....
(2) 022022 000004 TST47: SCOPE
2801
2802 022024 RRB1:
(1) 022024 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.
2803 022026 012737 022166 000004 MOV #10$,ERRVEC ; IN CASE IT TRAPS.
2804 022034 012700 022210 MOV #21$,R0
2805 022040 012701 000004 MOV #4,R1
2806 022044 012720 177777 1$: MOV #-1,(R0)+ ; NULL RECEIVING BUFFER.
2807 022050 077103 SOB R1,1$
2808
2809 022052 012737 022076 001164 MOV #2$,STMP2
2810 022060 013700 022200 MOV 20$,R0
2811 022064 170100 LDFPS R0 ; SET FPS.
2812 022066 012700 022210 MOV #21$,R0 ; SET FDST.
2813 022072 010037 001166 MOV R0,STMP3 ; SAVE AS R0 BEFORE.
2814 022076 170210 2$: STFPS (R0) ; TEST INSTRUCTION.
2815 022100 000241 241
2816 022102 020037 001166 CMP R0,STMP3 ; R0 SHOULD BE UNCHANGED.
2817 022106 001407 BEQ 3$ ; BR IF SO.
2818 022110 013737 001166 001170 MOV STMP3,STMP4
2819 022116 010037 001172 MOV R0,STMP5
2820 022122 104133 ERROR 133 ; R0 WRONG.
2821 022124 000417 BR 4$
2822
2823 022126 004537 004062 3$: JSR R5,CHECK4 ; CHECK STORED DATA.
2824 022132 022200 022210 20$,21$
2825 022136 001412 BEQ 4$
2826 022140 013737 022200 001166 MOV 20$,STMP3 ; FPS WORD.
2827 022146 012737 022200 001170 MOV #20$,STMP4 ; EXP'D.
2828 022154 012737 022210 001172 MOV #21$,STMP5 ; REC'D.
2829 022162 104134 ERROR 134 ; STORED FPS WRONG.
2830 022164 000415 4$: BR RRB DONE
2831
2832 022166 011637 001164 10$: MOV (SP),STMP2 ; TEST INSTR TRAPPED.
2833 022172 022626 CMP (SP)+,(SP)+
2834 022174 104135 ERROR 135
2835 022176 000410 BR RRB DONE
2836
2837 022200 102345 20$: .WORD 102345 ; FPS WORD.
2838 022202 177777 177777 177777 .WORD -1,-1,-1
2839 022210 177777 21$: .WORD -1 ; SHOULD GET STORED HERE...
2840 022212 177777 177777 177777 .WORD -1,-1,-1 ; ...BUT NOT HERE TOO.
2841
2842 022220 RRB DONE:
(1) 022220 104412 CLRFPS ;; CLEAR FP STATUS...
(3) 022222 000400 BR TST50 ;...AND PROCEED.
    
```

```
2844 .....  
3) *TEST 50 STFPS -- DST MODE 2  
(4) *  
(4) * TEST FDST MODE 2, USING THE STFPS INSTRUCTION.  
(4) *  
(3) .....  
(2) 022224 000004 TST50: SCOPE  
2845  
2846 022226 SSB1:  
(1) 022226 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.  
2847 022230 012737 022370 000004 MOV #10$,ERRVEC ; SET TRAP CATCHER.  
2848 022236 012700 022412 MOV #21$,R0  
2849 022242 012701 000004 MOV #4,R1  
2850 022246 012720 177777 1$: MOV #-1,(R0)+ ; NULL RECEIVING BUFFER.  
2851 022252 077103 SOB R1,1$  
2852  
2853 022254 012737 022300 001164 MOV #2$,STMP2  
2854 022262 013700 022402 MOV 20$,R0  
2855 022266 170100 LDFPS R0 ; SET FPS.  
2856 022270 012700 022412 MOV #21$,R0 ; SET FDST.  
2857 022274 010037 001166 MOV R0,STMP3 ; SAVE IT.  
2858 022300 170220 2$: STFPS (R0)+ ; TEST INSTRUCTION.  
2859 022302 000241 241  
2860  
2861 022304 020027 022414 CMP R0,#21$+2 ; AUTO-INCR OK ??  
2862 022310 001407 BEQ 3$ ; BR IF SO.  
2863 022312 012737 022414 001170 MOV #21$+2,STMP4  
2864 022320 010037 001172 MOV R0,STMP5  
2865 022324 104136 ERROR 136 ; RO WRONG.  
2866 022326 000417 BR 4$  
2867  
2868 022330 004537 004062 3$: JSR R5,CHECK4 ; CHECK RECEIVED DATA.  
2869 022334 022402 022412 20$,21$  
2870 022340 001412 BEQ 4$  
2871 022342 013737 022402 001166 MOV 20$,STMP3  
2872 022350 012737 022402 001170 MOV #20$,STMP4  
2873 022356 012737 022412 001172 MOV #21$,STMP5  
2874 022364 104137 ERROR 137 ; DATA STORED WRONG.  
2875 022366 000415 4$: BR SSBDONE  
2876  
2877 022370 011637 001164 10$: MOV (SP),STMP2 ; TEST INSTR TRAPPED.  
2878 022374 022626 CMP (SP)+,(SP)+  
2879 022376 104140 ERROR 140  
2880 022400 000410 BR SSBDONE  
2881  
2882 022402 105412 20$: .WORD 105412 ; FPS WORD.  
2883 022404 177777 177777 177777 .WORD -1,-1,-1  
2884 022412 177777 21$: .WORD -1 ; SHOULD GET STORED HERE.  
2885 022414 177777 177777 177777 .WORD -1,-1,-1  
2886  
2887 022422 SSBDONE:  
(1) 022422 104412 CLRFPs ;: CLEAR FP STATUS...  
(3) 022424 000400 BR TST51 ;:...AND PROCEED.
```

```

2889 .....
(3) : *TEST 51          STFPS -- DST MODE 4
(4) : *
(4) : * TEST FDST MODE 4, USING THE STFPS INSTRUCTION.
(4) : *
(3) : .....
(2) 022426 000004 TST51: SCOPE
2890
2891 022430 TTBI:
(1) 022430 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 - 1.
2892 022432 012737 022572 000004 MOV #10$,ERRVEC
2893 022440 012700 022614 MOV #21$,R0
2894 022444 012701 000004 MOV #4,R1
2895 022450 012720 177777 1$: MOV #-1,(R0)+ ; NULL THE BUFFER.
2896 022454 077103 SOB R1,1$
2897
2898 022456 012737 022502 001164 MOV #2$, $TMP2
2899 022464 013700 022604 MOV 20$,R0
2900 022470 170100 LDFPS R0 ; LOAD FPS.
2901 022472 012700 022616 MOV #21$+2,R0 ; SET FDST.
2902 022476 010037 001166 MOV R0,$TMP3 ; SAVE IT.
2903 022502 170240 2$: STFPS -(R0) ; TEST INSTRUCTION.
2904 022504 000241 241
2905
2906 022506 020027 022614 CMP R0,#21$ ; AUTO-DECR OK ??
2907 022512 001407 BEQ 3$ ; YES..
2908 022514 012737 022614 001170 MOV #21$, $TMP4
2909 022522 010037 001172 MOV R0,$TMP5
2910 022526 104141 ERROR 141 ; R0 WRONG.
2911 022530 000417 BR 4$
2912
2913 022532 004537 004062 3$: JSR R5,CHECK4 ; CHECK DATA BUFFER.
2914 022536 022604 022614 20$,21$
2915 022542 001412 BEQ 4$ ; BR IF OK.
2916 022544 013737 022604 001166 MOV 20$, $TMP3
2917 022552 012737 022604 001170 MOV #20$, $TMP4
2918 022560 012737 022614 001172 MOV #21$, $TMP5
2919 022566 104142 ERROR 142 ; STORED WRONG.
2920 022570 000415 4$: BR TTBDONE
2921
2922 022572 011637 001164 10$: MOV (SP), $TMP2 ; INSTRUCTION TRAPPED.
2923 022576 022626 CMP (SP)+,(SP)+
2924 022600 104143 ERROR 143
2925 022602 000410 BR TTBDONE
2926
2927 022604 105555 20$: .WORD 105555 ; FPS WORD.
2928 022606 177777 177777 177777 .WORD -1,-1,-1
2929 022614 177777 21$: .WORD -1 ; SHOULF END UP HERE.
2930 022616 177777 177777 177777 .WORD -1,-1,-1
2931
2932 022624 TTBDONE:
(1) 022624 104412 CLRFPs ;: CLEAR FP STATUS...
(3) 022626 000400 BR TST52 ;: ...AND PROCEED.
  
```

```

2934 .....
(3) *TEST 52          STFPS -- DST MODE 3
(4) *
(4) * TEST FDST MODE 3, USING THE STFPS INSTRUCTION.
(4) *
(3) .....
(2) 022630 000004 TST52: SCOPE
2935
2936 022632 UUB1:
(1) 022632 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
2937 022634 012737 022774 000004 MOV #10$,ERRVEC ; SET TRAP CATCHER.
2938 022642 012700 023016 MOV #21$,R0
2939 022646 012701 000004 MOV #4,R1
2940 022652 012720 177777 1$: MOV #-1,(R0)+ ; NULL BUFFER.
2941 022656 077103 SOB R1,1$
2942
2943 022660 012737 022704 001164 MOV #2$,STMP2
2944 022666 013700 023006 MOV 20$,R0
2945 022672 170100 LDFPS R0 ; LOAD UP FPS.
2946 022674 012700 023026 MOV #22$,R0 ; SET FDST.
2947 022700 010037 001166 MOV R0,$TMP3 ; SAVE IT.
2948 022704 170230 2$: STFPS @ (R0)+ ; TEST INSTRUCTION.
2949 022706 000241 241
2950
2951 022710 020027 023030 CMP R0,#22$+2 ; AUTO INCR OK ??
2952 022714 001407 BEQ 3$ ; YES.
2953 022716 012737 023030 001170 MOV #22$+2,$TMP4
2954 022724 010037 001172 MOV R0,$TMP5
2955 022730 104144 ERROR 144 ; R0 WRONG.
2956 022732 000417 BR 4$
2957
2958 022734 004537 004062 3$: JSR R5,CHECK4 ; CHECK DATA RECEIVED.
2959 022740 023006 023016 20$,21$
2960 022744 001412 BEQ 4$ ; BR IF OK.
2961 022746 013737 023006 001166 MOV 20$,STMP3
2962 022754 012737 023006 001170 MOV #20$,STMP4
2963 022762 012737 023016 001172 MOV #21$,STMP5
2964 022770 104145 ERROR 145 ; DATA WRONG.
2965 022772 000416 4$: BR UUBDONE
2966
2967 022774 011637 001164 10$: MOV (SP),$TMP2 ; INSTRUCTION TRAPPED.
2968 023000 022626 CMP (SP)+,(SP)+
2969 023002 104146 ERROR 146
2970 023004 000411 BR UUBDONE
2971
2972 023006 106653 20$: .WORD 106653 ; FPS WORD.
2973 023010 177777 177777 177777 .WORD -1,-1,-1
2974 023016 177777 21$: .WORD -1 ; SHOULD END UP HERE.
2975 023020 177777 177777 177777 .WORD -1,-1,-1
2976 023026 023016 22$: .WORD 21$ ; INDIRECT POINTER.
2977
2978 023030 UUBDONE:
(1) 023030 104412 CLRFPS ;: CLEAR FP STATUS...
(3) 023032 000400 BR TST53 ;: ...AND PROCEED.
  
```

```

2980
(3)
(4)
(4)
(4)
(3)
(2) 023034 000004
2981
2982 023036
(1) 023036 104411
2983 023040 012737 023200 000004
2984 023046 012700 023224
2985 023052 012701 000004
2986 023056 012720 177777
2987 023062 077103
2988
2989 023064 012737 023110 001164
2990 023072 013700 023212
2991 023076 170100
2992 023100 012700 023236
2993 023104 010037 001166
2994 023110 170250
2995 023112 000241
2996 023114 020027 023234
2997 023120 001407
2998 023122 012737 023234 001170
2999 023130 010037 001172
3000 023134 104147
3001 023136 000417
3002
3003 023140 004537 004062
3004 023144 023212 023224
3005 023150 001412
3006 023152 013737 023212 001166
3007 023160 012737 023212 001170
3008 023166 012737 023224 001172
3009 023174 104150
3010 023176 000417
3011
3012 023200 011637 001164
3013 023204 022626
3014 023206 104151
3015 023210 000412
3016
3017 023212 004301
3018 023214 177777 177777 177777
3019 023224 177777
3020 023226 177777 177777 177777
3021 023234 023224
3022
3023 023236
(1) 023236 104412
(3) 023240 000400

```

```

*****
*TEST 53          STFPS -- DST MODE 5
*
* TEST FDST MODE 5, USING THE STFPS INSTRUCTION.
*
*****
TST53:  SCOPE
VVB1:
LUPERR
MOV #10$,ERRVEC ;: LOOP HERE ON ERROR IF SWR9 = 1.
MOV #21$,R0 ;: SET TRAP CATCHER.
MOV #4,R1
1$: MOV #-1,(R0)+ ; NULL BUFFER.
SOB R1,1$
MOV #2$,STMP2
MOV 20$,R0
LDFPS R0 ;: LOAD FPS WORD.
MOV #22$+2,R0 ;: SET FDST.
MOV R0,STMP3 ;: AND SAVE IT.
2$: STFPS @-(R0) ;:TEST INSTRUCTION.
241
CMP R0,#22$ ;: AUTO-DECR OK ??
BEQ 3$
MOV #22$,STMP4 ;: NO
MOV R0,STMP5
MOV ERROR 147 ;: R0 WRONG.
BR 4$
3$: JSR R5,CHECK4 ;: CHECK DATA BUFFER.
20$,21$
BEQ 4$ ;: BR IF OK.
MOV 20$,STMP3
MOV #20$,STMP4
MOV #21$,STMP5
MOV ERROR 150 ;: STORED WRONG.
4$: BR VVBDONE
10$: MOV (SP),STMP2 ;: TEST INSTRUCTION TRAPPED.
CMP (SP)+,(SP)+
MOV ERROR 151
BR VVBDONE
20$: .WORD 004301 ;: FPS WORD.
3018 023214 177777 177777 177777
3019 023224 177777
3020 023226 177777 177777 177777
21$: .WORD -1,-1,-1, ;: SHOULD GO HERE.
3021 023234 023224
22$: .WORD -1,-1,-1 ;: INDIRECT POINTER.
21$
VVBDONE:
CLRFP
BR TST54 ;: CLEAR FP STATUS...
;:...AND PROCEED.

```

```

3025 .....
(3) *TEST 54 STFPS -- DST MODE 6
(4) *
(4) * TEST FDST MODE 6, USING THE STFPS INSTRUCTION.
(4) *
(3) .....
(2) 023242 000004 TST54: SCOPE
3026
3027 023244 WWB1:
(1) 023244 104411 LUPERR ;; LOOP HERE ON ERROR IF SWR9 = 1.
3028 023246 012737 023410 000004 MOV #10$,ERRVEC ; SET TRAP CATCHER.
3029 023254 012700 023432 MOV #21$,R0
3030 023260 012701 000004 MOV #4,R1
3031 023264 012720 177777 1$: MOV #-1,(R0)+ ; NULL BUFFER.
3032 023270 077103 SOB R1,1$
3033
3034 023272 012737 023316 001164 MOV #2$,STMP2
3035 023300 013700 023422 MOV 20$,R0
3036 023304 170100 LDFPS R0 ; SET FPS WORD.
3037 023306 012700 023171 MOV #21$-241,R0 ; SET FDST.
3038 023312 010037 001166 MOV R0,STMP3 ; SAVE IT.
3039 023316 170260 000241 2$: STFPS 241(R0) ; TEST INSTRUCTION.
3040 023322 000241 241
3041 023324 020037 001166 CMP R0,STMP3 ; R0 SHOULD BE UNCHANGED.
3042 023330 001407 BEQ 3$ ; BR IF SO.
3043 023332 013737 001166 001170 MOV STMP3,STMP4
3044 023340 010037 001172 MOV R0,STMP5
3045 023344 104152 ERROR 152 ; R0 WRONG.
3046 023346 000417 BR 4$
3047
3048 023350 004537 004062 3$: JSR R5,CHECK4 ; CHECK RECEIVED DATA.
3049 023354 023422 023432 20$,21$
3050 023360 001412 BEQ 4$ ; BR IF OK.
3051 023362 013737 023422 001166 MOV 20$,STMP3
3052 023370 012737 023422 001170 MOV #20$,STMP4
3053 023376 012737 023432 001172 MOV #21$,STMP5
3054 023404 104153 ERROR 153
3055 023406 000415 4$: BR WWBDONE
3056
3057 023410 011637 001164 10$: MOV (SP),STMP2 ; TEST INSTRUCTION TRAPPED.
3058 023414 022626 CMP (SP)+,(SP)+
3059 023416 104154 ERROR 154
3060 023420 000410 BR WWBDONE
3061
3062 023422 102514 20$: .WORD 102514 ; FPS WORD.
3063 023424 177777 177777 177777 .WORD -1,-1,-1
3064 023432 177777 21$: .WORD -1 ; GOES HERE.
3065 023434 177777 177777 177777 .WORD -1,-1,-1
3066
3067 023442 WWBDONE:
(1) 023442 104412 CLRFPS ;; CLEAR FP STATUS...
(3) 023444 000400 BR TST55 ;;...AND PROCEED.
  
```

```
3069 .....  
(3) *TEST 55 STFPS -- DST MODE 7  
(4) *  
(4) * TEST FDST MODE 7, USING THE STFPS INSTRUCTION.  
(4) *  
(7) .....  
(2) 023446 000004 TST55: SCOPE  
3070  
3071 023450 YYB1:  
(1) 023450 104411 LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.  
3072 023452 012737 023614 000004 MOV #10$,ERRVEC ; TRAP CATCHER.  
3073 023460 012700 023636 MOV #21$,R0  
3074 023464 012701 000004 MOV #4,R1  
3075 023470 012720 177777 1$: MOV #-1,(R0)+ ; NULL BUFFER.  
3076 023474 077103 SOB R1,1$  
3077  
3078 023476 012737 023522 001164 MOV #2$, $TMP2  
3079 023504 013700 023626 MOV 20$,R0  
3080 023510 170100 LDFPS R0 ; SET FPS.  
3081 023512 012700 023405 MOV #22$-241,R0 ; SET FDST.  
3082 023516 010037 001166 MOV R0,$TMP3 ; SAVE IT.  
3083 023522 170270 000241 2$: STFPS @241(R0) ; TEST INSTRUCTION.  
3084 023526 000241 241  
3085 023530 020037 001166 CMP R0,$TMP3 ; R0 SHOULD BE UNCHANGED.  
3086 023534 001407 BEQ 3$ ; BR IF SO.  
3087 023536 013737 001166 001170 MOV $TMP3,$TMP4  
3088 023544 010037 001172 MOV R0,$TMP5  
3089 023550 104155 ERROR 155 ; R0 WRONG.  
3090 023552 000417 BR 4$ 4$  
3091  
3092 023554 004537 004062 3$: JSR R5,CHECK4 ; DATA OK ??  
3093 023560 023626 023636 20$,21$  
3094 023564 001412 BEQ 4$ ; BR IF SO.  
3095 023566 013737 023626 001166 MOV 20$, $TMP3  
3096 023574 012737 023626 001170 MOV #20$, $TMP4  
3097 023602 012737 023636 001172 MOV #21$, $TMP5  
3098 023610 104156 ERROR 156 ; STORED WRONG.  
3099 023612 000416 4$: BR YYBDONE  
3100  
3101 023614 011637 001164 10$: MOV (SP), $TMP2 ; TEST INSTRUCTION TRAPPED.  
3102 023620 022626 CMP (SP)+,(SP)+  
3103 023622 104157 ERROR 157  
3104 023624 000411 BR YYBDONE  
3105  
3106 023626 103747 20$: .WORD 103747 ; FPS WORD.  
3107 023630 177777 177777 177777 .WORD -1,-1,-1  
3108 023636 177777 21$: .WORD -1 ; GOES HERE.  
3109 023640 177777 177777 177777 .WORD -1,-1,-1  
3110 023646 023636 22$: .WORD 21$ ; INDIRECT POINTER.  
3111  
3112 023650 YYBDONE:  
(1) 023650 104412 CLRFPSP ;: CLEAR FP STATUS...  
(3) 023652 000400 BR TST56 ;: ...AND PROCEED.
```


3119
(3)
(4)
(4)
(4)
(3)
(2) 023654 000004
3120
3121
3122
3123
3124 023656
(1) 023656 104411
3125 023660 170011
3126 023662 170012
3127 023664 012737 023676 001164
3128 023672 012700 024016
3129 023676 175420
3130 023700 000241
3131 023702 020027 024022
3132 023706 001411
3133
3134 023710 012737 024016 001166
3135 023716 012737 024022 001170
3136 023724 010037 001172
3137 023730 104160
3138
3139 023732
(1) 023732 104411
3140 023734 170011
3141 023736 170012
3142 023740 012737 023752 001164
3143 023746 012700 024016
3144 023752 175440
3145 023754 000241
3146 023756 020027 024012
3147 023762 001421
3148
3149 023764 012737 024016 001166
3150 023772 012737 024012 001170
3151 024000 010037 001172
3152 024004 104161
3153 024006 000407
3154
3155 024010 177777 177777 177777
3156 024016 177777
3157 024020 177777 177777 177777
3158
3159 024026
(1) 024026 104412
(3) 024030 000400

```
*****
: *TEST 56          STCDL -- DST MODES 2 AND 4
: *
: * TEST FDST MODES 2 AND 4, USING THE STCDL INSTRUCTION.
: *
: *****
TST56: SCOPE
:
: ALL WE CARE ABOUT HERE IS THE AUTO-INCREMENT/DECREMENT.
: WE SET FL - 1, TO TRY AND BREAK IT.
:
ZZB1:
      LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
      SETD           ; DOUBLE...
      SETL           ; ...AND LONG.
      MOV            #1$, $TMP2
      MOV            #10$, R0          ; SET FDST.
      STCDL         AC0, (R0)+       ; TEST INSTRUCTION.
      241
      CMP            R0, #10$+4      ; AUTO-INCR RIGHT ??
      BEQ            2$              ; YES, PROCEED.

      MOV            #10$, $TMP3
      MOV            #10$+4, $TMP4
      MOV            R0, $TMP5
      ERROR         160              ; AUTO-INCR FAILS

2$:
      LUPERR          ;; LOOP HERE ON ERROR IF SWR9 = 1.
      SETD
      SETL
      MOV            #3$, $TMP2
      MOV            #10$, R0        ; SET FDST.
      STCDL         AC0, -(R0)      ; TEST IT.
      241
      CMP            R0, #10$-4      ; AUTO-DECR OK ??
      BEQ            ZZBDONE        ; DONE IF SO.

      MOV            #10$, $TMP3
      MOV            #10$-4, $TMP4
      MOV            R0, $TMP5
      ERROR         161              ; AUTO-DECR FAILS.
      BR            ZZBDONE

10$:
      .WORD          -1, -1, -1
      .WORD          -1
      .WORD          -1, -1, -1

ZZBDONE:
      CLRFP
      BR            TST57           ;; CLEAR FP STATUS...
                                   ; ...AND PROCEED.
```

3167
(3)
(4)
(4)
(4)
(4)
(3)
(2) 024032 000004
3168
3169
3170
3171 024034 004737 024746
3172 024040 020000 000000 000000
3173 024050 000000 000000
3174 024054 040317
3175 024056 040304 000000
3176 024062 046104
3177
3178
3179
3180 024064 004737 024746
3181 024070 040000 000000 000000
3182 024100 000000 000000
3183 024104 040317
3184 024106 040304 000000
3185 024112 046104
3186
3187
3188
3189 024114 004737 024746
3190 024120 047667 075757 157737
3191 024130 055675 173757
3192 024134 040717
3193 024136 040700 000000
3194 024142 046104
3195
3196
3197
3198 024144 004737 024746
3199 024150 050000 000000 000000
3200 024160 000000 000000
3201 024164 040717
3202 024166 140705 000006
3203 024172 046104
3204
3205
3206
3207 024174 004737 024746
3208 024200 050000 000000 000000
3209 024210 000000 000000
3210 024214 040317
3211 024216 040305 000000
3212 024222 046104
3213
3214
3215

```
*****
*TEST 57          STCDI/STCDL AND STCFI/STCFL -- DST MODE 1
*
* TEST THE 'STORE CONVERTED' INSTRUCTIONS STCDI, STCDL
* STCFI, AND STCFL.
*****
TST57: SCOPE
:
: DOUBLE (EXPON = 100) TO LONG INTEGER.
:
AAAB1: JSR      PC,AAABSUB
:WORD      20000,0,0,0      ;ACO OPERAND.
:WORD      0,0              ;EXPECTED RESULT.
:WORD      40317           ; FID, FD, AND FL.
:WORD      40304,0        ; EXPECTED FPS FEC AFTER.
:ASCII     'DL'           ; ADJUST ERROR TEXT.
:
: DOUBLE (EXPON = 0) TO LONG INTEGER.
:
AAAB2: JSR      PC,AAABSUB
:WORD      40000,0,0,0      ;ACO OPERAND.
:WORD      0,0              ;EXPECTED RESULT.
:WORD      40317           ; FID, FD, AND FL.
:WORD      40304,0        ; FPS, FEC.
:ASCII     'DL'
:
: DOUBLE (EXPON = 37) TO LONG INTEGER, WITH FIC = 1.
:
AAAB3: JSR      PC,AAABSUB
:WORD      47667,75757,157737,167773 ;ACO OPERAND.
:WORD      55675,173757      ;EXPECTED RESULT.
:WORD      40717             ; FIC - 1.
:WORD      40700,0          ; FPS, FEC.
:ASCII     'DL'
:
: DOUBLE (EXP = 40) TO LONG WITH FIC = 1.
:
AAAB4: JSR      PC,AAABSUB
:WORD      50000,0,0,0      ;ACO OPERAND.
:WORD      0,0              ;EXPECTED RESULT.
:WORD      40717             ; FIC = 1.
:WORD      140705,6         ; FPS, FEC.
:ASCII     'DL'
:
: DOUBLE (EXP = 40) TO LONG WITH FIC = 0.
:
AAAB5: JSR      PC,AAABSUB
:WORD      50000,0,0,0      ;ACO OPERAND.
:WORD      0,0              ;EXPECTED RESULT.
:WORD      40317             ; FIC = 0.
:WORD      40305,0          ; FPS, FEC.
:ASCII     'DL'
:
: DOUBLE (EXP = 30) TO LONG WITH FIC = 1.
:
```

3216	024224	004737	024746		AAAB6: JSR PC,AAABSUB		
3217	024230	046000	000001	000000	.WORD 46000,1,0,0	:ACO OPERAND.	
3218	024240	000200	000001		.WORD 200,1	:EXPECTED RESULT.	
3219	024244	040717			.WORD 40717	: FIC = 1.	
3220	024246	040700	000000		.WORD 40700,0	: FPS, FEC.	
3221	024252	046104			.ASCII 'DL'		
3222							
3223					: DOUBLE (EXP = 27) TO LONG WITH FIC = 1.		
3224							
3225	024254	004737	024746		AAAB7: JSR PC,AAABSUB		
3226	024260	045600	000001	000000	.WORD 45600,1,0,0	:ACO OPERAND.	
3227	024270	000100	000000		.WORD 100,0	:EXPECTED RESULT.	
3228	024274	040717			.WORD 40717	: FIC = 1.	
3229	024276	040700	000000		.WORD 40700,0	: FPS, FEC.	
3230	024302	046104			.ASCII 'DL'		
3231							
3232					: DOUBLE (EXP = 17) TO SHORT WITH FIC = 1.		
3233							
3234	024304	004737	024746		AAAB8: JSR PC,AAABSUB		
3235	024310	043600	000000	000000	.WORD 43600,0,0,0	:ACO OPERAND.	
3236	024320	040000	177777		.WORD 40000,-1	:EXPECTED RESULT.	
3237	024324	040617			.WORD 40617	: FL = 0.	
3238	024326	040600	000000		.WORD 40600,0	: FPS, FEC.	
3239	024332	044504			.ASCII 'DI'		
3240							
3241					: DOUBLE (EXP = 20) TO SHORT WITH FIC = 1.		
3242							
3243	024334	004737	024746		AAAB9: JSR PC,AAABSUB		
3244	024340	044000	000000	000000	.WORD 44000,0,0,0	:ACO OPERAND.	
3245	024350	000000	177777		.WORD 0,-1	:EXPECTED RESULT.	
3246	024354	040617			.WORD 40617	: FL = 0.	
3247	024356	140605	000006		.WORD 140605,6	: FPS, FEC.	
3248	024362	044504			.ASCII 'DI'		
3249							
3250					: NEG DOUBLE (EXP = 10) TO SHORT WITH FIC = 1.		
3251							
3252	024364	004737	024746		AAAB10: JSR PC,AAABSUB		
3253	024370	142000	000000	000000	.WORD 142000,0,0,0	:ACO OPERAND.	
3254	024400	177600	177777		.WORD 177600,-1	:EXPECTED RESULT.	
3255	024404	040617			.WORD 40617	: FPS	
3256	024406	040610	000000		.WORD 40610,0	: FPS, FEC.	
3257	024412	044504			.ASCII 'DI'		
3258							
3259					: NEG DOUBLE (EXP = 37) TO LONG WITH FIC = 1.		
3260							
3261	024414	004737	024746		AAAB11: JSR PC,AAABSUB		
3262	024420	147600	000000	000000	.WORD 147600,0,0,0	:ACO OPERAND.	
3263	024430	140000	000000		.WORD 140000,0	:EXPECTED RESULT.	
3264	024434	040717			.WORD 40717	: FPS	
3265	024436	040710	000000		.WORD 40710,0	: FPS, FEC.	
3266	024442	046104			.ASCII 'DL'		
3267							
3268					: NEG DOUBLE (EXP = 37) TO LONG WITH FIC = 1.		
3269							
3270	024444	004737	024746		AAAB12: JSR PC,AAABSUB		
3271	024450	147600	000000	001000	.WORD 147600,0,1000,0	:ACO OPERAND.	

3272	024460	137777	177777		.WORD	137777,177777	:EXPECTED RESULT.
3273	024464	040717			.WORD	40717	:FPS
3274	024466	040710	000000		.WORD	40710,0	:FPS, FEC.
3275	024472	046104			.ASCII	'DL'	
3276							
3277							
3278							
3279	024474	004737	024746		AAAB13: JSR	PC,AAABSUB	
3280	024500	150200	000000	000000	.WORD	150200,0,0,0	:ACO OPERAND.
3281	024510	000000	000000		.WORD	0,0	:EXPECTED RESULT.
3282	024514	040717			.WORD	40717	:FPS
3283	024516	140705	000006		.WORD	140705,6	:FPS, FEC.
3284	024522	046104			.ASCII	'DL'	
3285							
3286							
3287							
3288	024524	004737	024746		AAAB14: JSR	PC,AAABSUB	
3289	024530	150000	000001	000000	.WORD	150000,1,0,0	:ACO OPERAND.
3290	024540	000000	000000		.WORD	0,0	:EXPECTED RESULT.
3291	024544	040717			.WORD	40717	:FPS
3292	024546	140705	000006		.WORD	140705,6	:FPS, FEC
3293	024552	046104			.ASCII	'DL'	
3294							
3295							
3296							
3297	024554	004737	024746		AAAB15: JSR	PC,AAABSUB	
3298	024560	150001	000000	000000	.WORD	150001,0,0,0	:ACO OPERAND.
3299	024570	000000	000000		.WORD	0,0	:EXPECTED RESULT.
3300	024574	040717			.WORD	40717	:FPS
3301	024576	140705	000006		.WORD	140705,6	:FPS, FEC.
3302	024602	046104			.ASCII	'DL'	
3303							
3304							
3305							
3306	024604	004737	024746		AAAB16: JSR	PC,AAABSUB	
3307	024610	150000	000000	000000	.WORD	150000,0,0,0	:ACO OPERAND.
3308	024620	100000	000000		.WORD	100000,0	:EXPECTED RESULT.
3309	024624	040717			.WORD	40717	:FPS
3310	024626	040710	000000		.WORD	40710,0	:FPS, FEC.
3311	024632	046104			.ASCII	'DL'	
3312							
3313							
3314							
3315	024634	004737	024746		AAAB17: JSR	PC,AAABSUB	
3316	024640	144000	000001	000000	.WORD	144000,1,0,0	:ACO OPERAND.
3317	024650	100000	177777		.WORD	100000,-1	:EXPECTED RESULT.
3318	024654	040617			.WORD	40617	:FPS
3319	024656	040610	000000		.WORD	40610,0	:FPS, FEC.
3320	024662	044504			.ASCII	'DI'	
3321							
3322							
3323							
3324	024664	004737	024746		AAAB18: JSR	PC,AAABSUB	
3325	024670	047777	177777	177777	.WORD	47777,-1,-1,-1	:AC
3326	024700	077777	177600		.WORD	77777,177600	:EXP RESULT.
3327	024704	040517			.WORD	40517	:FPS

3328 024706 040500 000000
 3329 024712 046106
 3330
 3331
 3332
 3333 024714 004737 024746
 3334 024720 047777 177777 177777
 3335 024730 000000 177777
 3336 024734 040417
 3337 024736 140405 000006
 3338 024742 044506
 3339
 3340 024744 000544
 3341
 3342
 3343
 3344
 3345
 3346
 3347
 3348
 3349
 3350
 3351
 3352
 3353
 3354
 3355
 3356
 3357 024746 012637 001164
 3358 024752 104411
 3359 024754 012700 177777
 3360 024760 012701 025240
 3361 024764 012702 000004
 3362 024770 010021
 3363 024772 077202
 3364
 3365 024774 013701 001164
 3366 025000 170011
 3367 025002 010100
 3368 025004 172410
 3369 025006 010037 001166
 3370 025012 062700 000010
 3371 025016 010037 001170
 3372 025022 062700 000006
 3373 025026 010037 001174
 3374
 3375 025032 016100 000014
 3376 025036 170100
 3377 025040 012700 025240
 3378 025044 000277
 3379 025046 175410
 3380
 3381 025050 106737 025254
 3382 025054 170200
 3383 025056 010037 025250

```

: SINGLE (EXP = 37) TO SHORT WITH FIC = 1.
AAAB19: JSR PC,AAABSUB
: WORD 47777,-1,-1,-1 : AC
: WORD 0,-1 : EXP RESULT
: WORD 40417 : FPS
: WORD 140405.6 : FPS, FEC.
: ASCII 'FI'

BR AAABDONE

: SUBROUTINE TO SET-UP, EXECUTE, AND CHECK RESULTS OF
: THE 'STORE CONVERTED' INSTRUCTIONS STCXX.

CALL:
: JSR PC,AAABSUB
: WORD X,X,X,X : AC OPERAND
: WORD X,X : EXPECTED RESULT
: WORD X : FPS BEFORE EXECUTION
: WORD X,X : EXP FPS, FEC AFTER.
: ASCII 'FI' : ERROR TEXT ADJUSTMENT.
: RETURN TO CALL+24

: NOTE THAT ON ERROR, THE 'ERROR PC' REPORTED IS THAT OF THE
: CALLING SEQUENCE AND NOT THAT OF THE CONVERT INSTRUCTION.

AAABSUB: MOV (SP)+,$TMP2 : SAVE CALL PC AS ERROR PC.
: LUPERR : LOOP HERE ON ERROR IF SWR9 1.
MOV #-1,R0
MOV #20$,R1
MOV #4,R2
1$: MOV R0,(R1)+ : NULL RECEIVING BUFFER.
SOB R2,1$

MOV $TMP2,R1 : SET ARG POINTER.
SETD
MOV R1,R0 : LOAD ACO.
LDD (R0),ACO
MOV R0,$TMP3 : SAVE POINTER TO AC...
ADD #10,R0
MOV R0,$TMP4 : ...EXP RESULT..
ADD #6,R0
MOV R0,$TMP6 : ...AND EXP STATUS.

MOV 14(R1),R0 : INITIALIZE THE FPS.
LDFPS R0
MOV #20$,R0 : SET FDST.
SCC : SET PSW CC BITS.
2$: STCDL ACO,(R0) : XCT STORE CONVERTED.

MFPS 22$ : SAVE PSW AFTER STCXX
STFPS R0
MOV R0,21$ : GET FPS...

```

3384	025062	005000				CLR	R0	
3385	025064	005761	000020			TST	20(R1)	;...IF EXP'D FEC IS NON-ZERO...
3386	025070	001401				BEQ	3\$	
3387	025072	170300				STST	R0	;...GET FEC TOO.
3388	025074	010037	025252		3\$:	MOV	R0,21\$+2	
3389								
3390	025100	013737	001170	025112		MOV	\$TMP4,4\$; SET EXP RESULT ADDRESS...
3391	025106	004537	004052			JSR	R5,CHECK2	;...AND CHECK RESULTING DATA.
3392	025112	000000	025240		4\$:		0,20\$	
3393	025116	001010				BNE	6\$	
3394	025120	013737	001174	025132		MOV	\$TMP6,5\$; SET EXP STATUS ADDRESS...
3395	025126	004537	004052			JSR	R5,CHECK2	;...AND CHECK RESULTING STATUS.
3396	025132	000000	025250		5\$:		0,21\$	
3397	025136	001412				BEQ	7\$	
3398								
3399	025140	016137	000022	043540	6\$:	MOV	22(R1),EM162X	; ADJUST ERROR TEXT.
3400	025146	012737	025240	001172		MOV	#20\$, \$TMP5	
3401	025154	012737	025250	001176		MOV	#21\$, \$TMP7	
3402	025162	104162				ERROR	162	; FP STATUS OR RESULT WRONG.
3403								
3404	025164	013746	025250		7\$:	MOV	21\$,-(SP)	; FPS...
3405	025170	042716	177760			BIC	#^C17,(SP)	;...CC BITS...
3406	025174	013746	025254			MOV	22\$,-(SP)	;...SHOULD HAVE BEEN COPIED...
3407	025200	042716	177760			BIC	#^C17,(SP)	;...INTO THE PSW.
3408	025204	022626				CMPL	(SP)+,(SP)+	
3409	025206	001412				BEQ	8\$	
3410	025210	016137	000022	043766		MOV	22(R1),EM165X	; ADJUST ERROR TEXT.
3411	025216	013737	025250	001166		MOV	21\$, \$TMP3	
3412	025224	013737	025254	001170		MOV	22\$, \$TMP4	
3413	025232	104165				ERROR	165	; PSW INCORRECT AFTER STCXX.
3414								
3415	025234	000161	000024		8\$:	JMP	24(R1)	; RETURN.
3416								
3417	025240	177777	177777	177777	20\$:	.WORD	-1,-1,-1,-1	; RECEIVE CONVERTED DATA.
3418	025250	177777	177777		21\$:	.WORD	-1,-1	; RECEIVED FPS AND FEC.
3419	025254	177777			22\$:	.WORD	-1	; PSW AFTER STCXX.
3420								
3421	025256					AAABDONE:		
(1)	025256	104412				CLRFPS		:: CLEAR FP STATUS...
(3)	025260	000400				BR	TST60	::...AND PROCEED.

3429
(3)
(4)
(4)
(4)
(4)
(3)
(2) 025262 000004
3430
3431
3432
3433 025264 004737 025442
3434 025270 020000 000000 000000
3435 025300 177700
3436 025302 040017 040010
3437
3438
3439
3440 025306 004737 025442
3441 025312 040000 000000 000000
3442 025322 000000
3443 025324 040017 040004
3444
3445
3446
3447 025330 004737 025442
3448 025334 040200 000000 000000
3449 025344 000001
3450 025346 040017 040000
3451
3452
3453
3454 025352 004737 025442
3455 025356 077200 000000 000000
3456 025366 000175
3457 025370 040017 040000
3458
3459
3460
3461 025374 004737 025442
3462 025400 000200 000000 000000
3463 025410 177601
3464 025412 040017 040010
3465
3466
3467
3468 025416 004737 025442
3469 025422 033400 000000 000000
3470 025432 177756
3471 025434 040017 040010
3472
3473 025440 000526
3474
3475
3476
3477

```
*****  
: TEST 60 STEXP -- DST MODE 0 AND 1  
: *  
: * TEST THE STEXP INSTRUCTION WITH A VARIETY OF OPERANDS.  
: * BOTH DST MODES 0 AND 1 ARE TESTED FOR EACH OPERAND SET.  
: *  
: *****  
TST60: SCOPE  
: EXP - 100 (EXCESS 200)  
: :  
: CCCB1: JSR PC,CCCBSUB ; XCT STEXP DST MODE 0 AND 1.  
: .WORD 20000,0,0,0 ; AC  
: .WORD -100 ; EXP EXPONENT.  
: .WORD 40017,40010 ; FPS BEFORE AND AFTER.  
: :  
: EXP = 200 (EXCESS 200)  
: :  
: CCCB2: JSR PC,CCCBSUB ;  
: .WORD 40000,0,0,0 ; ACO OPERAND.  
: .WORD 0 ; EXPECTED EXPONENT RESULT.  
: .WORD 40017,40004 ; FPS'S.  
: :  
: EXP = 201 (EXCESS 200)  
: :  
: CCCB3: JSR PC,CCCBSUB ;  
: .WORD 40200,0,0,0 ; ACO OPERAND.  
: .WORD 1 ; EXPECTED EXPONENT RESULT.  
: .WORD 40017,40000 ; FPS'S  
: :  
: EXP - 375 (EXCESS 200)  
: :  
: CCCB4: JSR PC,CCCBSUB ;  
: .WORD 77200,0,0,0 ; ACO OPERAND.  
: .WORD 175 ; EXPECTED EXPONENT RESULT.  
: .WORD 40017,40000 ; FPS'S.  
: :  
: EXP = 1 (EXCESS 200)  
: :  
: CCCB5: JSR PC,CCCBSUB ;  
: .WORD 200,0,0,0 ; ACO OPERAND.  
: .WORD -177 ; EXPECTED EXPONENT RESULT.  
: .WORD 40017,40010 ; FPS'S.  
: :  
: EXP - 156 (EXCESS 200)  
: :  
: CCCB6: JSR PC,CCCBSUB ;  
: .WORD 33400,0,0,0 ; ACO OPERAND.  
: .WORD -22 ; EXPECTED EXPONENT RESULT.  
: .WORD 40017,40010 ; FPS'S.  
: :  
BR CCCBDONE  
: SUBROUTINE TO EXECUTE AND CHECK THE STEXP INSTRUCTION.  
: CALL :
```

```

3478
3479
3480
3481
3482
3483
3484
3485
3486 025442 012637 001164
3487 025446 012737 175000 025564
3488 025454 112737 000177 043636
3489 025462 112737 000177 043641
3490 025470 000411
3491 025472 052737 000010 025564 7$:
3492 025500 112737 000050 043636
3493 025506 112737 000051 043641
3494
3495 025514 1$:
(1) 025514 104411
3496 025516 170011
3497 025520 013701 001164
3498 025524 010100
3499 025526 172410
3500 025530 010037 001172
3501 025534 016137 000010 001174
3502 025542 016100 000012
3503 025546 170100
3504 025550 016137 000014 001170
3505
3506 025556 012700 025710
3507 025562 000277
3508 025564 175000 2$:
3509 025566 106737 025714
3510 025572 105737 025564
3511 025576 001002
3512 025600 010037 025710
3513
3514 025604 170205 3$:
3515 025606 010537 001166
3516 025612 023737 001166 001170
3517 025620 001004
3518 025622 023737 025710 001174
3519 025630 001404
3520 025632 013737 025710 001176 4$:
3521 025640 104163
3522
3523 025642 013746 001166 5$:
3524 025646 042716 177760
3525 025652 013746 025714
3526 025656 042716 177760
3527 025662 022626
3528 025664 001404
3529 025666 013737 025714 001170
3530 025674 104166
3531
3532 025676 105737 025564 6$:
    
```

```

JSR PC,CCCBSUB
.WORD X,X,X,X ; AC OPERAND.
.WORD X ; EXPECTED EXPONENT.
.WORD X,X ; FPS BEFORE AND AFTER.
; RETURN TO CALL+16
    
```

SAVE THE CALL PC AS 'ERROR PC' IN CASE OF ERROR.

```

CCCBSUB: MOV (SP)+,$TMP2 ; SAVE AS 'ERROR PC'.
MOV #STEXP,2$ ; 1ST PASS, INIT OPCODE...
MOVB #177,EM163X
MOVB #177,EM163X+3 ;...AND TEXT TO 'STEXP ACO,R0'
BR 1$
7$: BIS #10,2$ ; 2ND PASS, CHANGE OPCODE...
MOVB #'(,EM163X
MOVB #''),EM163X+3 ;...AND TEXT TO 'STEXP ACO,(R0)''

1$: LUPERR ;: LOOP HERE ON ERROR IF SWR9 1.
SETD
MOV $TMP2,R1 ; GET POINTER.
MOV R1,R0
LDD (R0),ACO ; LOAD UP THE AC...
MOV R0,$TMP5 ;...AND SAVE POINTER.
MOV 10(R1),$TMP6 ; SAVE EXPECTED EXPONENT.
MOV 12(R1),R0
LDFPS R0 ; SET INITIAL FPS...
MOV 14(R1),$TMP4 ;...AND EXPECTED FPS.

MOV #10$,R0 ; NULL (MODE 0) OR DST (MODE 1).
SCC ; SET ALL PSW CC BITS.
2$: STEXP ACO,R0 ; XCT STEXP MODE 0 (AND MODE 1).
MFPS 11$ ; GET PSW AFTER.
TSTB 2$
BNE 3$ ; IF MODE 1, RESULT IS IN 10$...
MOV R0,10$ ;...OTHERWISE, PUT IT THERE NOW.

3$: STFPS R5 ;GET FPS.
MOV R5,$TMP3
CMP $TMP3,$TMP4 ; FPS RIGHT ??
BNE 4$ ; NO
CMP 10$,$TMP6 ; YES, STORED EXPONENT RIGHT ??
BEQ 5$ ; YES.

4$: MOV 10$,$TMP7
ERROR 163 ; EXPON OR FPS WRONG.

5$: MOV $TMP3,-(SP) ; FPS...
BIC #^C17,(SP) ;...CC BITS...
MOV 11$,-(SP) ;...SHOULD HAVE BEEN COPIED...
BIC #^C17,(SP) ;...INTO THE PSW.
CMP (SP)+,(SP)+
BEQ 6$
MOV 11$,$TMP4
ERROR 166 ; PSW INCORRECT AFTER STEXP.

6$: TSTB 2$ ; 1ST PASS (DST MODE 0) ??
    
```


3533	025702	001673		BEO	7\$: YES, GO 'ROUND AND DO MODE 1...
3534	025704	000161	000016	JMP	16(R1)	:...THEN RETURN.
3535						
3536	025710	177777	177777	10\$:	.WORD -1,-1	: SCRATCH BUFFER.
3537	025714	177777		11\$:	.WORD -1	: PSW AFTER STEXP.
3538						
3539	025716			CCCBDONE:		
(1)	025716	104412		CLRFPS		:: CLEAR FP STATUS...
(3)	025720	000400		BR	TST61	::...AND PROCEED.

```

3547
(3)
(4)
(4)
(4)
(4)
(3)
(2) 025722 000004
3548
3549 025724
(1) 025724 104411
3550 025726 012700 040000
3551 025732 170100
3552 025734 012700 026054
3553 025740 012710 177777
3554 025744 012760 177777 000002
3555 025752 012737 025760 001164
3556 025760 170017 1$: 170017
3557
3558
3559 025762 170310 2$: STST (R0)
3560 025764 000241 241
3561 025766 170200 STFPS R0
3562 025770 010037 001166 MOV R0,$TMP3
3563 025774 012737 140000 001170 MOV #140000,$TMP4
3564
3565 026002 023737 001166 001170 CMP $TMP3,$TMP4
3566 026010 001005 BNE 3$
3567 026012 004537 004052 JSR R5,CHECK2
3568 026016 026044 026054 10$,11$
3569 026022 001407 BEQ 4$
3570
3571 026024 012737 026044 001172 3$: MOV #10,$TMP5
3572 026032 012737 026054 001174 MOV #11,$TMP6
3573 026040 104164 ERROR 164
3574 026042 000410 4$: BR DDDBDONE
3575
3576 026044 000002 025760 10$: .WORD 2,1$
3577 026050 177777 177777 .WORD -1,-1
3578 026054 177777 177777 11$: .WORD -1,-1
3579 026060 177777 177777 .WORD -1,-1
3580
3581 026064 104412 DDDBDONE: CLRFPS
3582 026066 000137 026710 JMP $EOP
    
```

```

*****
*TEST 61 STST -- DST MODE 1
*
* TEST THE STST INSTRUCTION USING FDST MODE 1.
* VERIFY THAT THE RETURNED FEC AND FEA ARE CORRECT.
*
*****
TST61: SCOPE

DDDB1:
LUPERR ;: LOOP HERE ON ERROR IF SWR9 = 1.
MOV #40000,R0 ;:SET FPS. FID=1.
LDFPS R0
MOV #11$,R0 ;: SET FDST.
MOV #-1,(R0) ;: AND NULL FEC...
MOV #-1,2(R0) ;:...AND FEA RECEIVERS.
MOV #1$,$TMP2
1$: 170017 ;: XCT ILLEGAL FP OPCODE...
;:...TO SET FEC AND FEA.

2$: STST (R0) ;: NOW GET THEM.
241
STFPS R0 ;:GET FPS.
MOV R0,$TMP3
MOV #140000,$TMP4

CMP $TMP3,$TMP4 ;: FPS RIGHT ??
BNE 3$
JSR R5,CHECK2 ;: FEC/FEA RIGHT ??
10$,11$
BEQ 4$ ;: BR IF BOTH RIGHT.

3$: MOV #10,$TMP5
MOV #11,$TMP6
ERROR 164 ;: STATUS BAD.

4$: BR DDDBDONE

10$: .WORD 2,1$ ;: EXPECTED FEC AND FEA.
.WORD -1,-1
11$: .WORD -1,-1 ;: RECEIVED FEC AND FEA.
.WORD -1,-1

DDDBDONE: CLRFPS ;: CLEAR...
JMP $EOP ;:...AND END IT HERE.
    
```

3598
 (3)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2)
 3599
 3600
 3601
 3602
 3603
 3604
 3605
 3606
 3607
 3608
 3609
 3610
 3611
 3612
 (1)
 (1)
 3613
 3614
 3615
 3616
 3617
 3618
 3619
 3620
 3621
 3622
 3623
 3624
 3625
 3626
 3627
 3628
 3629
 3630
 3631
 3632
 3633
 3634
 3635

026072 000004
 026074 012737 026230 000064
 026102 012737 000340 000066
 026110 112737 000340 177776
 026116 105777 153026
 026122 100375
 026124 005000
 026126 052777 000100 153014
 026134 105077 153012
 026140 105037 177776
 026144 005200
 026146 001376
 026150 042777 000100 152772
 026156 104401 026164
 026162 000420
 026224
 026224 000137 003312
 026230 022626
 026232 010037 026520
 026236 004537 026322
 026242 172037
 026244 042101 042104
 026250 004537 026322
 026254 173037
 026256 052523 042102
 026262 004537 026322
 026266 171037
 026270 052515 042114
 026274 004537 026322
 026300 174437
 026302 044504 042126
 026306 004537 026322
 026312 171437
 026314 047515 042104
 026320 000555

```

:*****
:TEST 62 INTERRUPTABILITY TEST
:* FPF11 INTERRUPTABILITY TEST
:* THIS TEST IS INCLUDED IN CASE THE FPF11 PRESENTS INTERRUPT
:* LATENCY PROBLEMS. AT THE PRESENT TIME, ALL FPF11 INSTRUCTIONS
:* ARE NOT INTERRUPTABLE. HOPEFULLY, THIS WON'T BE A PROBLEM.
:* IF IT TURNS OUT THAT LATENCY IS EXCESSIVE -- THE MICROCODE WILL
:* HAVE TO BE TWEAKED TO PROVIDE INTERRUPTABILITY.
:* THIS TEST WILL EXECUTE ADD, SUBD, MULD, DIVD, AND MODD OPCODES,
:* ATTEMPT TO INTERRUPT (ABORT) THEM VIA TTY INTERRUPT, AND REPORT
:* WETHER OR NOT THE INSTRUCTION WAS IN FACT INTERRUPTED.
:* NO SPECIAL EQUIPMENT (OTHER THAN THE CONSOLE TERMINAL) IS REQUIRED.
:* THE TEST IS NOT INCLUDED IN THE NORMAL TEST SEQUENCE. IF YOU
:* WANT TO RUN IT -- CHANGE THE SWR TO 000462 (LOOP ON TEST 62).
:*****
TST62: SCOPE

MOV #ZZZ,TPVEC ;SET UP VECTOR FOR THIS TEST
MOV #340,TPVEC+2
MOVB #340,PSW ; RAISE CPU TO 7.
1$: TSTB @STPS ; WAIT FOR READY.
BPL 1$
CLR RO
BIS #100,@STPS ; ENABLE INTERRUPT ON XMT-RDY.
CLRB @STPS ; SEND A CHAR...
CLRB PSW ;...AND LOWER THE CPU.
2$: INC RO ;COUNT UP WHILE WE WAIT.
BNE 2$ ;IF NO INTERRUPT BEFORE 0...
BIC #100,@STPS ;...DISABLE INTERRUPT...
TYPE ,65$ ;:TYPE ASCIZ STRING
BR ,64$ ;:GET OVER THE ASCIZ
;:65$: .ASCIZ 'NO TTY INTERRUPT -- CAN'T RUN'<CRLF>
64$: JMP START ;...AND FORGET IT !!

ZZZ: CMP (SP)+,(SP)+ ; OK, FIX STACK, CPU STAYS AT LEVEL 7...
MOV RO,ZTIMR ;...AND SAVE THE INTERVAL TIMER...
;... (AT 2 MEMORY CYCLES PER COUNT).
JSR R5,ZZZ1 ; NOW, EXECUTE ADD.
.WORD ADDD+37
.ASCII 'ADD'
JSR R5,ZZZ1 ; EXECUTE SUBD.
.WORD SUBD+37
.ASCII 'SUBD'
JSR R5,ZZZ1 ; EXECUTE MULD.
.WORD MULD+37
.ASCII 'MULD'
JSR R5,ZZZ1 ; EXECUTE DIVD
.WORD DIVD+37
.ASCII 'DIVD'
JSR R5,ZZZ1 ; EXECUTE MODD.
.WORD MODD+37
.ASCII 'MODD'
BR ZZZDONE ; DONE.

; SUBROUTINE TO EXECUTE THE INTERRUPT TEST.

```

```

3636
3637 026322 012737 026434 000064 ZZZ1: MOV #ZZZ3,TPVEC ; SET VECTOR.
3638 026330 012537 026422 MOV (R5)+,ZZZ2 ; SET OP CODE.
3639 026334 012700 026544 MOV #ZFUNC+2,R0
3640 026340 012520 MOV (R5)+,(R0)+ ; AND TEXT.
3641 026342 012520 MOV (R5)+,(R0)+
3642 026344 013701 026520 MOV ZTIMR,R1 ; GET THE TIMER.
3643 026350 010100 ZZZ1A: MOV R1,R0 ; COPY DELAY TIMER => R0.
3644 026352 170127 000200 LDFPS #200 ; DOUBLE, CLEAR ERRORS AND INTERRUPTS.
3645 026356 172437 026522 LDD ZOP1,ACO ; LOAD UP THE AC OPERAND.
3646 026362 105777 152562 1$: TSTB @STPS ; INSURE HE'S READY...
3647 026366 100375 BPL 1$
3648 026370 052777 000100 152552 BIS #100,@STPS ;...AND INTERRUPT ENABLED.
3649 026376 105077 152550 CLRB @STPB ; SEND GO-AHEAD CHARACTER...
3650 026402 105037 177776 CLRB PSW ;...AND INTERRUPT 2K CYCLES FROM NOW.
3651 026406 005300 2$: DEC R0 ; NOTE THAT THIS LOOP IS LONGER THAN...
3652 026410 100404 BMI ZZZ2 ;...THAT USED TO SET THE TIMER IN THE...
3653 026412 000775 BR 2$ ;...1ST PLACE, GUARANTEING THAT THE...
3654 026414 000240 000240 240,240 ;...INTERRUPT WILL OCCUR BEFORE THE...
3655 026420 000777 BR . ;...MAX DELAY (R0 = ZTIMR) TIMES OUT.
3656
3657 026422 172037 026532 ZZZ2: ADDD ZOP2,ACO ; XCT TEST INSTRUCTION...
3658 026426 000240 000240 240,240 ;...IF IT FINISHES...
3659 026432 000777 BR . ;...STALL HERE 'TIL INTERRUPT OCCURS.
3660
3661 026434 042777 000100 152506 ZZZ3: BIC #100,@STPS ; OK, INTERRUPT RECEIVED, TURN IT OFF.
3662 026442 012600 MOV (SP)+,R0 ; SAVE THE PC...
3663 026444 005726 TST (SP)+ ;...AND DISCARD THE PS.
3664 026446 020027 026422 CMP R0,#ZZZ2
3665 026452 101010 BHI 1$ ; INTERRUPTED AFTER FUNCTION.
3666 026454 001414 BEQ 2$ ; INTERRUPTED DURING FUNCTION.
3667 026456 000240 NOP ; INTERRUPTED BEFORE FUNCTION STARTED...
3668 026460 077145 SOB R1,ZZZ1A ;...START-UP DELAY IS TOO LONG...
3669 ;...DECREASE IT AND TRY AGAIN.
3670 026462 104401 026542 TYPE ,ZFUNC ; IF WE REACH ZERO...
3671 026466 104401 026623 TYPE ,ZNGS ;...TEST FUNCTION NEVER STARTED ???
3672 026472 000411 BR 3$
3673 026474 104401 026542 1$: TYPE ,ZFUNC
3674 026500 104401 026574 TYPE ,ZWNI ; TEST FUNCTION WAS NOT INTERRUPTED.
3675 026504 000404 BR 3$
3676 026506 104401 026542 2$: TYPE ,ZFUNC
3677 026512 104401 026551 TYPE ,ZWI ; TEST FUNCTION WAS INTERRUPTED.
3678 026516 000205 3$: RTS R5
3679
3680 026520 000000 ZTIMR: 0 ; INTERRUPT TIMER.
3681 026522 041766 164570 152375 ZOP1: .FLT4 123.456 ; AC OPERAND.
3682 026532 040511 007717 100334 ZOP2: .FLT4 3.14159 ; FSRC OPERAND.
3683 026542 020040 042101 042104 ZFUNC: .ASCIZ ' ADDD' ; OR ONE OF THE OTHERS.
3684 026551 040 040527 020123 ZWI: .ASCIZ ' WAS INTERRUPTED.'<CRLF>
3685 026574 053440 051501 047040 ZWNI: .ASCIZ ' WAS NOT INTERRUPTED.'<CRLF>
3686 026623 040 042516 042526 ZNGS: .ASCIZ ' NEVER GOT STARTED ???'<CRLF>
3687 026654 .EVEN
3688
3689 026654 042777 000100 152266 ZZZDONE: BIC #100,@STPS ; DONE, INSURE TP INTERRUPT OFF.
3690 026662 012737 000066 000064 MOV #TPVEC+2,TPVEC ; RESET THE VECTOR.
3691 026670 005037 000066 CLR TPVEC+2
  
```

3692 026674 105037 177776
3693 026700 104401 001231
3694 026704 104412
(2) 026706 000400
3695
3696 000062

CLRB PSW ; LOWER THE CPU.
TYPE ,SRLF
CLRFPS ;; CLEAR FP STATUS...
BR \$EOP ;...AND SIGNAL END-PASS

LASTST= \$TN-1 ; REMEMBER LAST TEST NUMBER.

3698										
3699										
(1)										
(2)										
(1)										
(1)										
(1)										
(1)										
(1)										
(1)										
(1)										
(1)	026710									
(1)	026710	000004								
(1)	026712	005037	001102							
(1)	026716	005037	001220							
(1)	026722	005237	001242							
(1)	026726	042737	100000	001242						
(1)	026734	005327								
(1)	026736	000001								
(1)	026740	003063								
(1)	026742	012737								
(1)	026744	000001								
(1)	026746	026736								
(2)	026750	104401	026756							
(2)	026754	000407								
(2)										
(2)	026774									
(2)	026774	013746	001242							
(2)										
(2)	027000	104405								
(2)	027002	104401	027010							
(2)	027006	000421								
(2)										
(2)	027052									
(2)	027052	013746	001112							
(2)										
(2)	027056	104405								
(1)	027060	104401	001231							
(1)	027064	005037	001112							
(1)	027070	013700	000042							
(1)	027074	001405								
(1)	027076	000005								
(1)	027100	004710								
(1)	027102	000240								
(1)	027104	000240								
(1)	027106	000240								
(1)	027110									
(1)	027110	000137								
(1)	027112	003716								
(1)	027114	377	377	000						
(1)		027120								

```

.SBTTL
.SBTTL  END OF PASS ROUTINE

:*****
:*INCREMENT THE PASS NUMBER ($PASS)
:*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
:*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYYY'
:*WHERE XXXXX AND YYYYYY ARE DECIMAL NUMBERS
:*IF THERES A MONITOR GO TO IT
:*IF THERE ISN'T JUMP TO LOOP

$EOP:
      SCOPE
      CLR      $TSTNM      ;;ZERO THE TEST NUMBER
      CLR      $TIMES     ;;ZERO THE NUMBER OF ITERATIONS
      INC      $PASS      ;;INCREMENT THE PASS NUMBER
      BIC      #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
      DEC      (PC)+      ;;LOOP?

$EOPCT: .WORD 1
      BGT      $DOAGN     ;;YES
      MOV      (PC)+,@(PC)+ ;;RESTORE COUNTER

$ENDCT: .WORD 1
      $EOPCT
      TYPE     ,65$      ;;TYPE ASCIZ STRING
      BR       64$      ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
      MOV      $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
                        ;;TYPE PASS NUMBER
      TYPDS   ;;GO TYPE--DECIMAL ASCII WITH SIGN
      TYPE     ,67$      ;;TYPE ASCIZ STRING
      BR       66$      ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
      MOV      $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
                        ;;TOTAL NUMBER OF ERRORS
      TYPDS   ;;GO TYPE--DECIMAL ASCII WITH SIGN
      TYPE     , $CRLF   ;;TYPE CARRIAGE RETURN, LINE FEED
      CLR      $ERTTL   ;;CLEAR ERROR TOTAL
$GET42: MOV      @#42,R0  ;;GET MONITOR ADDRESS
      BEQ     $DOAGN   ;;BRANCH IF NO MONITOR
      RESET   ;;CLEAR THE WORLD
$ENDAD: JSR     PC,(R0)  ;;GO TO MONITOR
      NOP     ;;SAVE ROOM
      NOP     ;;FOR
      NOP     ;;ACT11

$DOAGN:
      JMP     @(PC)+    ;;RETURN
$RTNAD: .WORD  LOOP
$ENULL: .BYTE  -1,-1,0 ;;NULL CHARACTER STRING
      .EVEN
  
```

3701

(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)

(1) 027120
(1) 027120 104:07
(1) 027122 032777 0400C0 152010
(1) 027130 001133
(1) 027132 000416
(1) 027134 013746 000004
(1) 027140 012737 027160 000004
(1) 027146 005737 177060
(1) 027152 012637 000004
(1) 027156 000502
(1) 027160 022626
(1) 027162 012637 000004
(1) 027166 000442
(1) 027170
(1) 027170 032777 000400 151742
(1) 027176 001423
(1) 027200 005046
(1) 027202 117716 151732
(1) 027206 042716 000200
(1) 027212 001414
(2) 027214 022716 000062
(1) 027220 002411
(1) 027222 011637 001102
(1) 027226 005316
(1) 027230 006316
(1) 027232 062716 027436
(1) 027236 013637 001106
(1) 027242 000466
(1) 027244 005726
(1) 027246 105737 001103
(1) 027252 001421
(1) 027254 123737 001115 001103
(1) 027262 101015
(1) 027264 032777 001000 151646
(1) 027272 001404
(1) 027274 013737 001110 001106
(1) 027302 000446
(1) 027304 105037 001103
(1) 027310 005037 001220
(1) 027314 000415

```
.SBTTL SCOPE HANDLER ROUTINE
;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW14=1 LOOP ON TEST
;SW11=1 INHIBIT ITERATIONS
;SW09=1 LOOP ON ERROR
;SW08=1 LOOP ON TEST IN SWR<6:0>
;CALL
;* SCOPE ;:SCOPE=IOT

$SCOPE:
CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
;BNE $OVER ;:YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
;THIS INSTRUCTION TO A 'NOP' (NOP=240)
MOV @WERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
MOV #5$,@WERRVEC ;:SET FOR TIMEOUT
TST @177060 ;:TIME OUT ON XOR?
MOV (SP)+,@WERRVEC ;:RESTORE THE ERROR VECTOR
BR $$VLAD ;:GO TO THE NEXT TEST
5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+,@WERRVEC ;:RESTORE THE ERROR VECTOR
BR 7$ ;:LOOP ON THE PRESENT TEST
6$:;*****END OF CODE FOR THE XOR TESTER*****
BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
BEQ 2$ ;:BR IF NO
CLR -(SP) ;:CLEAR A TEMP. LOCATION
MOVB @SWR,(SP) ;:PICKUP THE DESIRED TEST NUMBER
BIC #SSWRMK,(SP) ;:MASK OUT UNDESIRED BITS
BEQ 8$ ;:BRANCH IF BAD TEST NUMBER IN SWR
CMP #62,(SP) ;:CHECK THE NUMBER IN THE SWR
BLT 8$ ;:BRANCH IF TEST NUMBER IS OUT OF RANGE
MOV (SP), $TSTNM ;:UPDATE THE TEST NUMBER
DEC (SP) ;:BACKUP BY ONE
ASL (SP) ;:SCALE THE TEST NUMBER AS AN INDEX
ADD #SSW08TBL,(SP) ;:FORM THE ADDRESS OF TEST POINTER
MOV @(SP)+,$LPADR ;:SET LOOP ADDRESS TO DESIRED TEST
BR $OVER ;:GO LOOP ON THE TEST
8$: TST (SP)+ ;:CLEAN THE BAD TEST NUMBER OFF OF THE STACK
2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
BEQ 3$ ;:BR IF NO
CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 3$ ;:BR IF NO
BIT #BIT09,@SWR ;:LOOP ON ERROR?
BEQ 4$ ;:BR IF NO
7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
BR $OVER
4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
CLR $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 1$ ;:ESCAPE TO THE NEXT TEST
```

```

(1) 027316 032777 004000 151614 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
(1) 027324 001011 BNE 1$ ;;BR IF YES
(1) 027326 005737 001242 TST $PASS ;;IF FIRST PASS OF PROGRAM
(1) 027332 001406 BEQ 1$ ;; INHIBIT ITERATIONS
(1) 027334 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
(1) 027340 023737 001220 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
(1) 027346 002024 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
(1) 027350 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
(1) 027356 013737 027434 001220 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
(1) 027364 105237 001102 $SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS
(1) 027370 113737 001102 001240 MOVB $TSTNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
(1) 027376 011637 001106 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
(1) 027402 011637 001110 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
(1) 027406 005037 001222 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1) 027412 112737 000001 001115 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1) 027420 013777 001102 151514 $OVER: MOV $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1) 027426 013716 001106 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
(2) 027432 000463 BR SCOPEX
(1) 027434 000001 $MXCNT: 1 ;;MAX. NUMBER OF ITERATIONS
(1) 027436 $SWOBTBL:
(3) 027436 004130 .WORD TST1+2 ;;STARTING ADDRESS OF TEST 1
(3) 027440 004264 .WORD TST2+2 ;;STARTING ADDRESS OF TEST 2
(3) 027442 004502 .WORD TST3+2 ;;STARTING ADDRESS OF TEST 3
(3) 027444 005066 .WORD TST4+2 ;;STARTING ADDRESS OF TEST 4
(3) 027446 005306 .WORD TST5+2 ;;STARTING ADDRESS OF TEST 5
(3) 027450 005522 .WORD TST6+2 ;;STARTING ADDRESS OF TEST 6
(3) 027452 005750 .WORD TST7+2 ;;STARTING ADDRESS OF TEST 7
(3) 027454 006176 .WORD TST10+2 ;;STARTING ADDRESS OF TEST 10
(3) 027456 006406 .WORD TST11+2 ;;STARTING ADDRESS OF TEST 11
(3) 027460 006630 .WORD TST12+2 ;;STARTING ADDRESS OF TEST 12
(3) 027462 007536 .WORD TST13+2 ;;STARTING ADDRESS OF TEST 13
(3) 027464 007640 .WORD TST14+2 ;;STARTING ADDRESS OF TEST 14
(3) 027466 010074 .WORD TST15+2 ;;STARTING ADDRESS OF TEST 15
(3) 027470 010204 .WORD TST16+2 ;;STARTING ADDRESS OF TEST 16
(3) 027472 010314 .WORD TST17+2 ;;STARTING ADDRESS OF TEST 17
(3) 027474 010552 .WORD TST20+2 ;;STARTING ADDRESS OF TEST 20
(3) 027476 011070 .WORD TST21+2 ;;STARTING ADDRESS OF TEST 21
(3) 027500 011416 .WORD TST22+2 ;;STARTING ADDRESS OF TEST 22
(3) 027502 011744 .WORD TST23+2 ;;STARTING ADDRESS OF TEST 23
(3) 027504 012300 .WORD TST24+2 ;;STARTING ADDRESS OF TEST 24
(3) 027506 012636 .WORD TST25+2 ;;STARTING ADDRESS OF TEST 25
(3) 027510 013156 .WORD TST26+2 ;;STARTING ADDRESS OF TEST 26
(3) 027512 013500 .WORD TST27+2 ;;STARTING ADDRESS OF TEST 27
(3) 027514 013764 .WORD TST30+2 ;;STARTING ADDRESS OF TEST 30
(3) 027516 014252 .WORD TST31+2 ;;STARTING ADDRESS OF TEST 31
(3) 027520 014470 .WORD TST32+2 ;;STARTING ADDRESS OF TEST 32
(3) 027522 015372 .WORD TST33+2 ;;STARTING ADDRESS OF TEST 33
(3) 027524 015532 .WORD TST34+2 ;;STARTING ADDRESS OF TEST 34
(3) 027526 015674 .WORD TST35+2 ;;STARTING ADDRESS OF TEST 35
(3) 027530 016052 .WORD TST36+2 ;;STARTING ADDRESS OF TEST 36
(3) 027532 016234 .WORD TST37+2 ;;STARTING ADDRESS OF TEST 37
(3) 027534 016416 .WORD TST40+2 ;;STARTING ADDRESS OF TEST 40
(3) 027536 016562 .WORD TST41+2 ;;STARTING ADDRESS OF TEST 41
(3) 027540 016746 .WORD TST42+2 ;;STARTING ADDRESS OF TEST 42
(3) 027542 017070 .WORD TST43+2 ;;STARTING ADDRESS OF TEST 43
(3) 027544 017164 .WORD TST44+2 ;;STARTING ADDRESS OF TEST 44
    
```



```

(1) 030060 000770 BR 7$ ;;LOOP
(1)
(1) ;HORIZONTAL TAB PROCESSOR
(1) 030062 112716 000040 8$: MOVB #' (SP) ;;REPLACE TAB WITH SPACE
(1) 030066 004737 030106 9$: JSR PC,$TYPEC ;;TYPE A SPACE
(1) 030072 132737 000007 030224 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
(1) 030100 001372 BNE 9$ ;;TAB STOP
(1) 030102 005726 TST (SP)+ ;;POP SPACE OFF STACK
(1) 030104 000724 BR 2$ ;;GET NEXT CHARACTER
(1) 030106 $TYPEC:
(1) 030106 105777 151032 TSTB @STKS ;;CHAR IN KYBD BUFFER? ;:MJD001
(1) 030112 100022 BPL 10$ ;;BR IF NOT ;:MJD001
(1) 030114 017746 151026 MOV @STKB,-(SP) ;;GET CHAR ;:MJD001
(1) 030120 042716 177600 BIC #177600,(SP) ;;STRIP EXTRANEIOUS BITS ;:MJD001
(1) 030124 122716 000023 CMPB #$XOFF,(SP) ;;WAS CHAR XOFF ;:MJD001
(1) 030130 001012 BNE 102$ ;;BR IF NOT ;:MJD001
(1) 030132 101$:
(1) 030132 105777 151006 TSTB @STKS ;;WAIT FOR CHAR ;:MJD001
(1) 030136 100375 BPL 101$ ;:MJD001
(1) 030140 117716 151002 MOVB @STKB,(SP) ;;GET CHAR ;:MJD001
(1) 030144 042716 177600 BIC #177600,(SP) ;;STRIP IT ;:MJD001
(1) 030150 122716 000021 CMPB #$XON,(SP) ;;WAS IT XON? ;:MJD001
(1) 030154 001366 BNE 101$ ;;BR IF NOT ;:MJD001
(1) 030156 102$:
(1) 030156 005726 TST (SP)+ ;;FIX STACK ;:MJD001
(1) 030160 10$:
(1) 030160 105777 150764 TSTB @STPS ;;WAIT UNTIL PRINTER IS READY ;:MJD001
(1) 030164 100375 BPL 10$ ;:MJD001
(1) 030166 116677 000002 150756 MOVB 2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 030174 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
(1) 030202 001003 BNE 1$ ;;BRANCH IF NO
(1) 030204 105037 030224 (LRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
(1) 030210 000406 BR $TYPEX ;;EXIT
(1) 030212 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
(1) 030220 001402 BEQ $TYPEX ;;BRANCH IF YES
(1) 030222 105227 INCB (PC)+ ;;COUNT THE CHARACTER
(1) 030224 000000 $CHARCNT: WORD 0 ;;CHARACTER COUNT STORAGE
(1) 030226 000207 $TYPEX: RTS PC
(1)
3732 ;////////////////////////////////////
3733 ; NOW BACK UP AND PATCH $TYPEC TO ENABLE <^G> RECOGNITION
3734 ; DURING OUTPUT SEQUENCES.
3735 ;
3736 ; SVPC= ; SAVE THE PC...
3737 ; $TYPEC+50 ; ...AND POINT TO 102$.
3738 030156 000424 102$: BR $$TCX ; BR TO PATCH...
3739 030160 10$: ; ...AND RETURN.
3740 030230 ; = SVPC ; RESTORE PC.
3741
3742 030230 122627 000007 $$TCX: CMPB (SP)+,#7 ; <^G> ???
3743 030234 001020 BNE 1$ ; RETURN IF NOT.
3744 ;.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
(1) 030236 005737 000042 TST @#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
(1) 030242 001012 BNE 64$ ;;BRANCH IF YES
(1) 030244 123727 001254 000001 CMPB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
    
```

```

(1) 030252 001406 BEQ 64$ ::BRANCH IF YES
(1) 030254 023727 001140 000176 CMP SWR,#SWREG ::SOFTWARE SWITCH REG SELECTED?
(1) 030262 001005 BNE 65$ ::BRANCH IF NO
(1) 030264 104406 GTSWR ::GET SOFT-SWR SETTINGS
(1) 030266 000403 BR 65$
(1) 030270 112737 000001 001134 64$: MOVB #1,$AUTOB ::SET AUTO-MODE INDICATOR
(1) 030276 1$: BR $TYPEC+52 : RETURN.
3745 030276 000730
3746 :////////////////////////

```

3748

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```
*****  
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT  
*OCTAL (ASCII) NUMBER AND TYPE IT.  
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE  
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPOS   ;;CALL FOR TYPEOUT  
*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE  
*   .BYTE  M              ;;M=1 OR 0  
*                               ;;1=TYPE LEADING ZEROS  
*                               ;;0=SUPPRESS LEADING ZEROS
```

```
*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST  
*$TYPOS OR $TYPOC
```

```
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPON   ;;CALL FOR TYPEOUT
```

```
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
```

```
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPOC   ;;CALL FOR TYPEOUT
```

```
(1) 030300 017646 000000  
(1) 030304 116637 000001 030523  
(1) 030312 112637 030525  
(1) 030316 062716 000002  
(1) 030322 000406  
(1) 030324 112737 000001 030523  
(1) 030332 112737 000006 030525  
(1) 030340 112737 000005 030522  
(1) 030346 010346  
(1) 030350 010446  
(1) 030352 010546  
(1) 030354 113704 030525  
(1) 030360 005404  
(1) 030362 062704 000006  
(1) 030366 110437 030524  
(1) 030372 113704 030523  
(1) 030376 016605 000012  
(1) 030402 005003  
(1) 030404 006105 1$:  
(1) 030406 000404 BR 3$  
(1) 030410 006105 2$:  
(1) 030412 006105 ROL R5  
(1) 030414 006105 ROL R5  
(1) 030416 010503 MOV R5,R3  
(1) 030420 006103 3$:  
(1) 030422 105337 030524 ROL R3  
(1) 030426 100016 DECB $OMODE  
(1) 030430 042703 177770 BPL 7$  
(1) 030434 001002 BIC #177770,R3  
(1) 030436 005704 BNE 4$  
(1) 030440 001403 TST R4  
BEQ 5$
```

```
$TYPOS: MOV @ (SP),-(SP) ;;PICKUP THE MODE  
MOV 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH  
MOV (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE  
ADD #2, (SP) ;;ADJUST RETURN ADDRESS  
BR $TYPON  
$TYPOC: MOV #1, $OFILL ;;SET THE ZERO FILL SWITCH  
MOV #6, $OMODE+1 ;;SET FOR SIX(6) DIGITS  
$TYPON: MOV #5, $OCNT ;;SET THE ITERATION COUNT  
MOV R3, -(SP) ;;SAVE R3  
MOV R4, -(SP) ;;SAVE R4  
MOV R5, -(SP) ;;SAVE R5  
MOV $OMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE  
NEG R4  
ADD #6, R4 ;;SUBTRACT IT FOR MAX. ALLOWED  
MOV R4, $OMODE ;;SAVE IT FOR USE  
MOV $OFILL, R4 ;;GET THE ZERO FILL SWITCH  
MOV 12(SP), R5 ;;PICKUP THE INPUT NUMBER  
CLR R3 ;;CLEAR THE OUTPUT WORD  
1$: ROL R5 ;;ROTATE MSB INTO 'C'  
BR 3$ ;;GO DO MSB  
2$: ROL R5 ;;FORM THIS DIGIT  
3$: ROL R3 ;;GET LSB OF THIS DIGIT  
DECB $OMODE ;;TYPE THIS DIGIT?  
BPL 7$ ;;BR IF NO  
BIC #177770, R3 ;;GET RID OF JUNK  
BNE 4$ ;;TEST FOR 0  
TST R4 ;;SUPPRESS THIS 0?  
BR 5$ ;;BR IF YES
```

(1)	030442	005204		4\$:	INC	R4	::DON'T SUPPRESS ANYMORE 0'S
(1)	030444	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
(1)	030450	052703	000040	5\$:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY
(1)	030454	110337	030520		MOVB	R3,8\$::SAVE FOR TYPING
(1)	030460	104401	030520		TYPE	,8\$::GO TYPE THIS DIGIT
(1)	030464	105337	030522	7\$:	DECB	\$OCNT	::COUNT BY 1
(1)	030470	003347			BGT	2\$::BR IF MORE TO DO
(1)	030472	002402			BLT	6\$::BR IF DONE
(1)	030474	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
(1)	030476	000744			BR	2\$::GO DO THE LAST DIGIT
(1)	030500	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
(1)	030502	012604			MOV	(SP)+,R4	::RESTORE R4
(1)	030504	012603			MOV	(SP)+,R3	::RESTORE R3
(1)	030506	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
(1)	030514	012616			MOV	(SP)+,(SP)	
(1)	030516	000002			RTI		::RETURN
(1)	030520	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
(1)	030521	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
(1)	030522	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
(1)	030523	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
(1)	030524	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE


```

(3) 030710 012601
(3) 030712 012600
(1) 030714 104401 030742
(1) 030720 016666 000002 000004
(1) 030726 012616
(1) 030730 000002
(1) 030732 023420
(1) 030734 001750
(1) 030736 000144
(1) 030740 000012
(1) 030742 000004

```

```

MOV (SP)+,R1      ;;POP STACK INTO R1
MOV (SP)+,R0      ;;POP STACK INTO R0
TYPE ,SDBLK       ;;NOW TYPE THE NUMBER
MOV 2(SP),4(SP)  ;;ADJUST THE STACK
MOV (SP)+,(SP)
RTI               ;;RETURN TO USER

```

```

$DTBL: 10000.
        1000.
        100.
        10.

```

```

$SDBLK: .BLKW 4

```

```

3751 :////////////////////////////////////////////////////
3752 : NOW BACK UP AND PATCH $TYPDS TO REALLY NULL LEAD ZEROS
3753 : INSTEAD OF REPLACING THEM WITH SPACES.
3754 :

```

```

3755 030752
3756 030636
3757 030636 062702 177661
3758 030642 062702 000177
3759 030752
3760

```

```

SVPC=      ; SAVE PC...
.= $TYPDS+110 ; ...AND POINT TO 6$.
6$: ADD #60-177,R2 ; CONVERT N TO ASCII N - NULL.
7$: ADD #177,R^ ; CONVERT 0 TO NULL.
.- SVPC ; RESTORE PC.

```

```

: ////////////////////////////////////////////////////

```


3762

.SBTTL APT COMMUNICATIONS ROUTINE

```
(1)
(2)
(1) 030752 112737 000001 031216 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
(1) 030760 112737 000001 031214 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
(1) 030766 000403 BR $ATYC
(1) 030770 112737 000001 031216 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
(1) 030776 $ATYC:
(3) 030776 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 031000 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(1) 031002 105737 031214 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
(1) 031006 001450 BEQ 5$ ;;IF NOT: BR
(1) 031010 122737 000001 001254 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
(1) 031016 001031 BNE 3$ ;;IF NOT: BR
(1) 031020 132737 000100 001255 BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 031026 001425 BEQ 3$ ;;IF NOT: BR
(1) 031030 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
(1) 031034 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 031042 005737 001234 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
(1) 031046 001375 BNE 1$ ;;IF NOT: WAIT
(1) 031050 010037 001250 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
(1) 031054 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 031056 001376 BNE 2$
(1) 031060 163700 001250 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 031064 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
(1) 031066 010037 001252 MOV R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
(1) 031072 012737 000004 001234 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
(1) 031100 000413 BR 5$
(1) 031102 017637 000004 031126 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
(1) 031110 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
(3) 031116 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
(1) 031122 004737 027674 JSR PC,$TYPE ;;CALL TYPE MACRO
(1) 031126 000000 4$: .WORD 0
(1) 031130 5$:
(1) 031130 105737 031216 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
(1) 031134 001416 BEQ 12$ ;;IF NOT: BR
(1) 031136 005737 001254 TST $ENV ;;RUNNING UNDER APT?
(1) 031142 001413 BEQ 12$ ;;IF NOT: BR
(1) 031144 005737 001234 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
(1) 031150 001375 BNE 11$ ;;IF NOT: WAIT
(1) 031152 017637 000004 001236 MOV @4(SP),$FATAL ;;GET ERROR #
(1) 031160 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 031166 005237 001234 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
(1) 031172 105037 031216 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
(1) 031176 105037 031215 CLRB $LFLG ;;CLEAR LOG FLAG
(1) 031202 105037 031214 CLRB $MFLG ;;CLEAR MESSAGE FLAG
(3) 031206 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
(3) 031210 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
(1) 031212 000207 RTS PC ;;RETURN
(1) 031214 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
(1) 031215 000 $LFLG: .BYTE 0 ;;LOG FLAG
(1) 031216 000 $FFLG: .BYTE 0 ;;FATAL FLAG
(1) 031220 .EVEN
(1) 000200 APTSIZE=200
(1) 000001 APTENV=001
(1) 000100 APTPOOL=100
```

(1) 000040 APTCSUP=040


```

(1) 031432 002420          BLT      18$          ;;BRANCH IF YES
(1) 031434 021627 000067  CMP      (SP),#67    ;;CHAR > 7?
(1) 031440 003015          BGT      18$          ;;BRANCH IF YES
(1) 031442 042726 000060  BIC      #60,(SP)+   ;;STRIP-OFF ASCII
(1) 031446 005766 000002  TST      2(SP)       ;;IS THIS THE FIRST CHAR
(1) 031452 001403          BEQ      17$          ;;BRANCH IF YES
(1) 031454 006316          ASL      (SP)        ;;NO, SHIFT PRESENT
(1) 031456 006316          ASL      (SP)        ;;CHAR OVER TO MAKE
(1) 031460 006316          ASL      (SP)        ;;ROOM FOR NEW ONE.
(1) 031462 005266 000002  17$: INC      2(SP)       ;;KEEP COUNT OF CHAR
(1) 031466 056616 177776  BIS      -2(SP),(SP) ;;SET IN NEW CHAR
(1) 031472 000707          BR       7$          ;;GET THE NEXT ONE
(1) 031474 104401 001230  18$: TYPE   ,SQUES    ;;TYPE ?<CR><LF>
(1) 031500 000720          BR       20$         ;;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) $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
(1) 031504 016666 000004 000002 MOV      4(SP),2(SP) ;;SAVE THE PS
(1) 031512 105777 147426 1$: TSTB    @STKS      ;;WAIT FOR
(1) 031516 100375          BPL      1$          ;;A CHARACTER
(1) 031520 117766 147422 000004 MOVB    @STKB,4(SP) ;;READ THE TTY
(1) 031526 042766 177600 000004 BIC      #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
(1) 031534 026627 000004 000023 CMP      4(SP),#23   ;;IS IT A CONTROL-S?
(1) 031542 001013          BNE      3$          ;;BRANCH IF NO
(1) 031544 105777 147374 2$: TSTB    @STKS      ;;WAIT FOR A CHARACTER
(1) 031550 100375          BPL      2$          ;;LOOP UNTIL ITS THERE
(1) 031552 117746 147370 MOVB    @STKB,-(SP) ;;GET CHARACTER
(1) 031556 042716 177600 BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
(1) 031562 022627 000021 CMP      (SP)+,#2    ;;IS IT A CONTROL-Q?
(1) 031566 001366          BNE      2$          ;;IF NOT DISCARD IT
(1) 031570 000750          BR       1$          ;;YES, RESUME
(1) 031572 026627 000004 000140 3$: CMP      4(SP),#140  ;;IS IT UPPER CASE?
(1) 031600 002407          BLT      4$          ;;BRANCH IF YES
(1) 031602 026627 000004 000175 CMP      4(SP),#175  ;;IS IT A SPECIAL CHAR?
(1) 031610 003003          BGT      4$          ;;BRANCH IF YES
(1) 031612 042766 000040 000004 BIC      #40,4(SP)   ;;MAKE IT UPPER CASE
(1) 031620 000002          RTI          ;;GO BACK TO USER
(1) 031622 052536 005015 000 $CNTLU: .ASCIZ / ^U/<15><12> ;;CONTROL 'U'
(1) 031627 136 006507 000012 $CNTLG: .ASCIZ / ^G/<15><12> ;;CONTROL 'G'
(1) 031634 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
(1) 031645 040 047040 053505 $MNEW: .ASCIZ / NEW - /
    
```

```

3766          .SBTTL TRAP DECODER
(1)
(2)          ;*****
(1)          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
(1)          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)          ;*GO TO THAT ROUTINE.
(1)
(1) 031656 010046          $TRAP: MOV     R0,-(SP)          ;;SAVE R0
(1) 031660 016600 000002  MOV     2(SP),R0          ;;GET TRAP ADDRESS
(1) 031664 005740          TST     -(R0)            ;;BACKUP BY 2
(1) 031666 111000          MOVB   (R0),R0          ;;GET RIGHT BYTE OF TRAP
(1) 031670 006300          ASL    R0                ;;POSITION FOR INDEXING
(1) 031672 016000 031712  MOV     $TRPAD(R0),R0    ;;INDEX TO TABLE
(1) 031676 000200          RTS    R0                ;;GO TO ROUTINE
(1)
(1)          ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
(1)
(1) 031700 011646          $TRAP2: MOV    (SP),-(SP) ;;MOVE THE PC DOWN
(1) 031702 016666 000004 000002 MOV    4(SP),2(SP)      ;;MOVE THE PSW DOWN
(1) 031710 000002          RTI                    ;;RESTORE THE PSW
(1)
(3)          .SBTTL TRAP TABLE
(3)          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)          ;*BY THE 'TRAP' INSTRUCTION.
(3)
(3)          :          ROUTINE
(3)          :          -----
(3) 031712 031700          $TRPAD: .WORD  $TRAP2
(3) 031714 027674          $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
(3) 031716 030324          $TYPOC ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3) 031720 030300          $TYPOS ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3) 031722 030340          $TYPON ;;CALL=TYPON       TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
(3) 031724 030526          $TYPDS ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(3) 031726 031270          $GTSWR ;;CALL=GTSWR     TRAP+6(104406) GET SOFT-SWR SETTING
(1)
(3) 031730 031220          $CKSWR ;;CALL=CKSWR     TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
(3) 031732 031502          $RDCHR ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
3767 031734 031740          .LPER  ;;CALL=LUPERR   TRAP+11(104411) SET LOOP-ON-ERROR ADDRESS
3768 031736 031746          .CLRFP ;;CALL=CLRFP      TRAP+12(104412) CLEAR FPS AND END TEST
3769          000026          $TERM=.-$TRPAD
3770
3771          ; THIS IS TO SET THE LOOP-ON-ERROR (SWR9) ADDRESS.
3772          ; CALLED VIA 'TRAP+11'.
3773
3774 031740 011637 001110  .LPER: MOV     (SP), $LPERR
3775 031744 000002          RTI
3776
3777          ; COME HERE AT END OF EACH TEST TO CLEAR FPS AND RESET VECTORS.
3778          ; CALLED VIA 'TRAP+12'.
3779
3780 031746 012700 032000  .CLRFP: MOV    #1$,R0
3781 031752 010037 000244  MOV    R0,FPVEC          ; JUST IN CASE THE...
    
```

3782	031756	010037	000010		MOV	RO,RESVEC	:...LDFPS IS BAD.
3783	031762	010037	000004		MOV	RO,ERRVEC	
3784	031766	005000			CLR	RO	
3785	031770	170100			LDFPS	RO	: CLEAR FP STATUS.
3786	031772	000240	000240		240,240		
3787	031776	000401			SKP1		
3788	032000	022626		1S:	CMP	(SP)+,(SP)+	: IF IT TRAPPED, JUST FIX THE STACK.
3789	032002	012737	003750	000244	MOV	#TRP244,FPVEC	: RESET ALL VECTORS.
3790	032010	012737	004004	000010	MOV	#TRP10,RESVEC	
3791	032016	012737	003774	000004	MOV	#TRP04,ERRVEC	
3792	032024	000002			RTI		

3794

.SBTTL POWER DOWN AND UP ROUTINES

```

(1)
(2)
(1)
(1) 032026 012737 032172 000024 $PWRDN: MOV    #SILLUP,@#PWRVEC ;;SET FOR FAST UP
(1) 032034 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
(3) 032042 010046                MOV    R0,-(SP)      ;;PUSH R0 ON STACK
(3) 032044 010146                MOV    R1,-(SP)      ;;PUSH R1 ON STACK
(3) 032046 010246                MOV    R2,-(SP)      ;;PUSH R2 ON STACK
(3) 032050 010346                MOV    R3,-(SP)      ;;PUSH R3 ON STACK
(3) 032052 010446                MOV    R4,-(SP)      ;;PUSH R4 ON STACK
(3) 032054 010546                MOV    R5,-(SP)      ;;PUSH R5 ON STACK
(3) 032056 017746 147056        MOV    @SWR,-(SP)    ;;PUSH @SWR ON STACK
(1) 032062 010637 032176        MOV    SP,$SAVR6    ;;SAVE SP
(1) 032066 012737 032100 000024  MOV    #SPWRUP,@#PWRVEC ;;SET UP VECTOR
(1) 032074 000000                HALT
(1) 032076 000776                BR     .-2          ;;HANG UP
(*)
(2)
(1)
(1) 032100 012737 032172 000024 $PWRUP: MOV    #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1) 032106 013706 032176        MOV    $SAVR6,SP    ;;GET SP
(1) 032112 005037 032176        CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
(1) 032116 005237 032176        1$:  INC    $SAVR6   ;;WAIT FOR THE INC
(1) 032122 001375                BNE    1$          ;;OF WORD
(3) 032124 012677 147010        MOV    (SP)+,@SWR   ;;POP STACK INTO @SWR
(3) 032130 012605                MOV    (SP)+,R5    ;;POP STACK INTO R5
(3) 032132 012604                MOV    (SP)+,R4    ;;POP STACK INTO R4
(3) 032134 012603                MOV    (SP)+,R3    ;;POP STACK INTO R3
(3) 032136 012602                MOV    (SP)+,R2    ;;POP STACK INTO R2
(3) 032140 012601                MOV    (SP)+,R1    ;;POP STACK INTO R1
(3) 032142 012600                MOV    (SP)+,R0    ;;POP STACK INTO R0
(1) 032144 012737 032026 000024  MOV    #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 032152 012737 000340 000026  MOV    #340,@#PWRVEC+2 ;;PRIO:7
(1) 032160 104401                TYPE                                ;;REPORT THE POWER FAILURE
(1) 032162 032742                $PWRMG: .WORD POWERM              ;;POWER FAIL MESSAGE POINTER
(1) 032164 012716                MOV    (PC)+,(SP)  ;;RESTART AT START
(1) 032166 003312                $PWRAD: .WORD START              ;;RESTART ADDRESS
(1) 032170 000002                RTI
(1) 032172 000000                $ILLUP: HALT
(1) 032174 000776                BR     .-2          ;;THE POWER UP SEQUENCE WAS STARTED
(1) 032176 000000                $SAVR6: 0           ;;BEFORE THE POWER DOWN WAS COMPLETE
;;PUT THE SP HERE
    
```



```

3798 .SBTTL ERROR TYPE OUT ROUTINE
3799
3800 : THIS ROUTINE IS CALLED FROM $ERROR, TO PRINT THE VARIOUS
3801 : ERROR SIGNATURES.
3802
3803 ERTYPE: MOV R0,-(SP) ;SAVE REGISTERS.
3804 MOV R1,-(SP)
3805 MOV R2,-(SP)
3806 MOV R3,-(SP)
3807 TYPE ,SCLF
3808 MOV $STNM,$TMP0
3809 BIC #^C377,$TMP0 ; SET TEST NUMBER...
3810 MOV $ERRPC,$TMP1 ;...AND CALL PC.
3811 MOV $ITEMB,R0 ; GET ERROR ITEM NUMBER.
3812 BIC #^C377,R0
3813 BEQ ERT4 ; JUST IGNORE ERROR 0.
3814
3815 2$: DEC R0 ; NOW MAKE INTO AN INDEX.
3816 ASL R0
3817 ASL R0
3818 ASL R0
3819 ADD # $ERRTB,R0 ; R0 POINTS TO 'ITEM XX'.
3820 MOV (R0)+,$3 ; GET EMXX POINTER.
3821 BEQ 4$
3822 TYPE ; TYPE EM TEXT.
3823 0
3824 3$: TYPE ,SCLF
3825 4$: MOV (R0)+,$5 ; GET DHXX POINTER.
3826 BEQ 6$
3827 TYPE ; TYPE DH TEXT.
3828 0
3829 5$: TYPE ,SCLF
3830
3831 6$: MOV (R0)+,R1 ; R1 POINTS TO 'DTXX'...
3832 MOV (R0),R0 ;...AND R0 TO 'DFXX'.
3833 ERT1: TST (R1)
3834 BEQ ERT4 ; EXIT AT END OF DATA TABLE.
3835
3836 1$: TSTB (R0) ; FORMAT 0 ??
3837 BNE 2$
3838 MOV @ (R1)+,-(SP) ; YES, TYPE AN OCTAL VALUE.
3839 TYPOC
3840 BR ERT2
3841
3842 2$: CMPB (R0),#2 ; FORMAT 2 ??
3843 BNE 3$
3844 MOV @ (R1)+,R2 ; YES, TYPE 2 OCTAL VALUES.
3845 MOV (R2)+,-(SP)
3846 TYPOC
3847 TYPE ,SPAC1
3848 MOV (R2),-(SP)
3849 TYPOC
3850 BR ERT2
3851
3852 3$: CMPB (R0),#3 ; FORMAT 3 ??
3853 BNE 5$
  
```

3854	032576	012703	000004		MOV	#4,R3		; YES, TYPE 4 OCTAL VALUES.
3855	032602	013102			MOV	@(R1)+,R2		
3856	032604	012246		4\$:	MOV	(R2)+,-(SP)		
3857	032606	104402			TYP0C			
3858	032610	104401	032736		TYPE	,SPAC1		
3859	032614	077305			SOB	R3,4\$		
3860	032616	000434			BR	ERT2		
3861								
3862	032620	121027	000004	5\$:	CMPB	(R0),#4		; FORMAT 4 ??
3863	032624	001004			BNE	6\$		
3864	032626	013146			MOV	@(R1)+,-(SP)		; YES, TYPE 4 OCTAL DIGITS...
3865	032630	104403			TYPOS			;...SUPPRESS LEAD ZEROS.
3866	032632	004	000		.BYTE	4,0		
3867	032634	000425			BR	ERT2		
3868								
3869	032636	121027	000005	6\$:	CMPB	(R0),#5		; FORMAT 5 ??
3870	032642	001005			BNE	10\$		
3871	032644	012137	032652		MOV	(R1)+,7\$; YES, TYPE ASCII STRING.
3872	032650	104401			TYPE			
3873	032652	000000		7\$:	0			
3874	032654	000417			BR	ERT3		
3875								
3876	032656	121027	000012	10\$:	CMPB	(R0),#12		; FORMAT 12 ??
3877	032662	001011			BNE	12\$		
3878	032664	012703	000010		MOV	#8,R3		; YES, TYPE 8 OCTAL VALUES.
3879	032670	013102			MOV	@(R1)+,R2		
3880	032672	012246		11\$:	MOV	(R2)+,-(SP)		
3881	032674	104402			TYP0C			
3882	032676	104401	032736		TYPE	,SPAC1		
3883	032702	077305			SOB	R3,11\$		
3884	032704	000401			BR	ERT2		
3885	032706	000000		12\$:	HALT			; UNDEFINED FORMAT CODE.
3886								
3887	032710	104401	032740	ERT2:	TYPE	,HTAB		; PRINT A TAB (AFTER DATA ONLY).
3888	032714	005200		ERT3:	INC	R0		; POINT TO NEXT FORMAT BYTE.
3889	032716	000702			BR	ERT1		
3890								
3891	032720	104401	001231	ERT4:	TYPE	,\$CRLF		
3892	032724	012603			MOV	(SP)+,R3		; RESTORE REGISTERS.
3893	032726	012602			MOV	(SP)+,R2		
3894	032730	012601			MOV	(SP)+,R1		
3895	032732	012600			MOV	(SP)+,R0		
3896	032734	000207			RTS	PC		
3897								
3898	032736	040	000	SPAC1:	.BYTE	40,0		
3899	032740	011	000	HTAB:	.BYTE	11,0		

```

3901 .SBTTL ASCII TEXT AND ERROR MESSAGES
3902
3903 ; FIRST SOME BITS AND PIECES COMMON TO MANY.
3904
3905 .NLIST BEX
3906 032742 050200 053517 051105 POWERM: .ASCIZ <CRLF>'POWER FAIL -- RESTARTING'<CRLF>
3907 032775 101 047502 052122 ABORT: .ASCIZ 'ABORTING TEST...'<CRLF>
3908 033017 200 054105 042520 ESTAT: .ASCIZ <CRLF>'EXPECTED FPS AND FEC: '
3909 033047 200 042522 042503 RSTAT: .ASCIZ <CRLF>'RECEIVED FPS AND FEC: '
3910 033077 200 020040 020040 RZB: .ASCIZ <CRLF>' (R0) BEFORE: '
3911 033122 042600 050130 020104 ERZA: .ASCIZ <CRLF>'EXPD (R0) AFTER: '
3912 033145 200 042522 042101 RRZA: .ASCIZ <CRLF>'READ (R0) AFTER: '
3913 033170 042600 050130 041505 EXP: .ASCIZ <CRLF>'EXPECTED: '
3914 033204 051200 041505 044505 RCD: .ASCIZ <CRLF>'RECEIVED: '
3915 033220 020200 024040 041501 ACZ: .ASCIZ <CRLF>' (AC0): '
3916 033234 020200 041501 047440 ACOPR: .ASCIZ <CRLF>' AC OPERAND: '
3917 033253 200 051123 020103 SOPR: .ASCIZ <CRLF>'SRC OPERAND: '
3918 033272 042200 052123 047440 DOPR: .ASCIZ <CRLF>'DST OPERAND: '
3919 033311 200 054105 042520 EXPX: .ASCIZ <CRLF>'EXPECTED EXPONENT: '
3920 033336 051200 041505 044505 RCDX: .ASCIZ <CRLF>'RECEIVED EXPONENT: '
3921 033363 200 054105 042520 ESTAT1: .ASCIZ <CRLF>'EXPECTED FEC AND FEA: '
3922 033413 200 042522 042503 RSTAT1: .ASCIZ <CRLF>'RECEIVED FEC AND FEA: '
3923
3924 ; THESE ARE ERROR MESSAGES:
3925
3926 033443 123 043124 040440 EM1: .ASCIZ 'STF ACO,AC7 DID NOT TRAP (AC7 IS ILLEGAL).'
3927 033516 052123 052101 051525 EM2: .ASCIZ 'STATUS INCORRECT AFTER STF ACO,AC7 TRAPPED.'
3928
3929 033572 042106 052123 024040 EM4: .ASCIZ 'FDST (R0) INCORRECT AFTER STF ACO,(R0).'
3930 033642 040504 040524 044440 EM5: .ASCIZ 'DATA INCORRECT AFTER STF ACO,(R0).'
3931 033705 123 043124 040440 EM6: .ASCIZ 'STF ACO,(R0) USED WRONG DATA MODE (BUT FD).'
3932 033761 106 051504 020124 EM7: .ASCIZ 'FDST (R0) INCORRECT AFTER ST'
3933 034015 106 040440 030103 EM7X: .ASCIZ 'F ACO,(R0)+'
3934 034031 104 052101 020101 EM10: .ASCIZ 'DATA INCORRECT AFTER ST'
3935 034060 020106 041501 026060 EM10X: .ASCIZ 'F ACO,(R0)+'
3936 034074 040504 040524 044440 EM11: .ASCIZ 'DATA INCORRECT AFTER STD ACO,(PC)+.'
3937 034140 042522 052524 047122 EM12: .ASCIZ 'RETURN PC INCORRECT AFTER STD ACO,(PC)+.'
3938 034211 123 042124 040440 EM13: .ASCIZ 'STD ACO,(PC)+ TRAPPED TO 4.'
3939 034245 106 051504 020124 EM14: .ASCIZ 'FDST (R0) INCORRECT AFTER STD ACO,-(R0).'
3940 034316 040504 040524 044440 EM15: .ASCIZ 'DATA INCORRECT AFTER STD ACO,-(R0).'
3941 034362 052123 020104 041501 EM16: .ASCIZ 'STD ACO,-(R0) TRAPPED TO 4.'
3942 034416 042106 052123 024040 EM17: .ASCIZ 'FDST (R0) INCORRECT AFTER STD ACO,@(R0)+.'
3943 034470 040504 040524 044440 EM20: .ASCIZ 'DATA INCORRECT AFTER STD ACO,@(R0)+.'
3944 034535 123 042124 040440 EM21: .ASCIZ 'STD ACO,@(R0)+ TRAPPED TO 4.'
3945 034572 042106 052123 024040 EM22: .ASCIZ 'FDST (R0) INCORRECT AFTER STD ACO,@-(R0).'
3946 034644 040504 040524 044440 EM23: .ASCIZ 'DATA INCORRECT AFTER STD ACO,@-(R0).'
3947 034711 123 042124 040440 EM24: .ASCIZ 'STD ACO,@-(R0) TRAPPED TO 4.'
3948 034746 042106 052123 024040 EM25: .ASCIZ 'FDST (R0) INCORRECT AFTER STD ACO,241(R0).'
3949 035021 104 052101 020101 EM26: .ASCIZ 'DATA INCORRECT AFTER STD ACO,241(R0).'
3950 035067 123 042124 040440 EM27: .ASCIZ 'STD ACO,241(R0) TRAPPED TO 4.'
3951 035125 106 051504 020124 EM30: .ASCIZ 'FDST (R0) INCORRECT AFTER STD ACO,@241(R0).'
3952 035201 104 052101 020101 EM31: .ASCIZ 'DATA INCORRECT AFTER STD ACO,@241(R0).'
3953 035250 052123 020104 041501 EM32: .ASCIZ 'STD ACO,@241(R0) TRAPPED TO 4.'
3954
3955 035307 103 047117 042526 EM34: .ASCIZ 'CONVERSION OR STATUS INCORRECT AFTER STC'<177>
3956 035357
  
```

3957		035360				.EVEN	
3958	035360	042106	040440	030103	EM34X:	.ASCIZ	'FD ACO,(RO).'
3959	035375	123	040524	052524	EM35:	.ASCIZ	'STATUS INCORRECT AFTER STCFD ACO,AC6 (ILLEGAL AC ERROR).'
3960	035466	042522	052523	052114	EM36:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER CLR'
3961	035527	106	024040	030122	EM36X:	.ASCIZ	'F (RO).'
3962	035537	123	040524	052524	EM37:	.ASCIZ	'STATUS INCORRECT AFTER CLRD AC7 (ILLEGAL AC ERROR).'
3963	035623	123	040524	052524	EM40:	.ASCIZ	'STATUS INCORRECT AFTER NEGD AC7 (ILLEGAL AC ERROR).'
3964	035707	122	051505	046125	EM41:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD ACO.'
3965	035757	106	051504	020124	EM42:	.ASCIZ	'FDST (RO) INCORRECT AFTER NEGD (RO).'
3966	036024	042522	052523	052114	EM43:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD (RO).'
3967	036075	116	043505	020104	EM44:	.ASCIZ	'NEGD (RO) TRAPPED TO 4.'
3968	036125	106	051504	020124	EM45:	.ASCIZ	'FDST (RO) INCORRECT AFTER ABSD (RO)+.'
3969	036173	122	051505	046125	EM46:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER ABSD (RO)+.'
3970	036245	101	051502	020104	EM47:	.ASCIZ	'ABSD (RO)+ TRAPPED TO 4.'
3971	036276	042106	052123	024040	EM50:	.ASCIZ	'FDST (RO) INCORRECT AFTER ABSD -(RO).'
3972	036344	042522	052523	052114	EM51:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER ABSD -(RO).'
3973	036416	041101	042123	026440	EM52:	.ASCIZ	'ABSD -(RO) TRAPPED TO 4.'
3974	036447	106	051504	020124	EM53:	.ASCIZ	'FDST (RO) INCORRECT AFTER NEGD @ (RO)+.'
3975	036516	042522	052523	052114	EM54:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD @ (RO)+.'
3976	036571	116	043505	020104	EM55:	.ASCIZ	'NEGD @ (RO)+ TRAPPED TO 4.'
3977	036623	106	051504	020124	EM56:	.ASCIZ	'FDST (RO) INCORRECT AFTER NEGD @-(RO).'
3978	036672	042522	052523	052114	EM57:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD @-(RO).'
3979	036745	116	043505	020104	EM60:	.ASCIZ	'NEGD @-(RO) TRAPPED TO 4.'
3980	036777	106	051504	020124	EM61:	.ASCIZ	'FDST (RO) INCORRECT AFTER ABSD 241(RO).'
3981	037047	122	051505	046125	EM62:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER ABSD 241(RO).'
3982	037123	101	051502	020104	EM63:	.ASCIZ	'ABSD 241(RO) TRAPPED TO 4.'
3983	037156	042106	052123	024040	EM64:	.ASCIZ	'FDST (RO) INCORRECT AFTER ABSD @241(RO).'
3984	037227	122	051505	046125	EM65:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER ABSD @241(RO).'
3985	037304	041101	042123	040040	EM66:	.ASCIZ	'ABSD @241(RO) TRAPPED TO 4.'
3986	037340	042522	052523	052114	EM67:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD ADDR (MODE 6, GR7).'
3987	037427	116	043505	020104	EM70:	.ASCIZ	'NEGD ADDR (MODE 6, GR7) TRAPPED TO 4.'
3988	037475	122	051505	046125	EM71:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER ABSD ADDR (MODE 7, GR7).'
3989	037564	041101	042123	040440	EM72:	.ASCIZ	'ABSD ADDR (MODE 7, GR7) TRAPPED TO 4.'
3990	037632	042522	052523	052114	EM73:	.ASCIZ	'RESULT OR FPS INCORRECT AFTER NEGD (PC)+.'
3991	037704	042522	052524	047122	EM74:	.ASCIZ	'RETURN PC INCORRECT AFTER NEGD (PC)+.'
3992	037752	042516	042107	024040	EM75:	.ASCIZ	'NEGD (PC)+ TRAPPED TO 4.'
3993	040003	122	051505	046125	EM76:	.ASCII	'RESULT OR STATUS INCORRECT AFTER '<177>'
3994		040044				.=-1	
3995						.EVEN	
3996	040044	042516	042107	024040	EM76X:	.ASCIZ	'NEGD (RO).'
3997							::: OR ABSD, OR TSTD.
3998	040057	106	051123	020103	EM100:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS (RO).'
3999	040125	106	051520	044440	EM101:	.ASCIZ	'FPS INCORRECT AFTER LDFPS (RO).'
4000	040165	114	043104	051520	EM102:	.ASCIZ	'LDFPS (RO) TRAPPED TO 4.'
4001	040216	051506	041522	024040	EM103:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS (RO)+.'
4002	040265	106	051520	044440	EM104:	.ASCIZ	'FPS INCORRECT AFTER LDFPS (RO)+.'
4003	040326	042114	050106	020123	EM105:	.ASCIZ	'LDFPS (RO)+ TRAPPED TO 4.'
4004	040360	051506	041522	024040	EM106:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS -(RO).'
4005	040427	106	051520	044440	EM107:	.ASCIZ	'FPS INCORRECT AFTER LDFPS -(RO).'
4006	040470	042114	050106	020123	EM110:	.ASCIZ	'LDFPS -(RO) TRAPPED TO 4.'
4007	040522	051506	041522	024040	EM111:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS @ (RO)+.'
4008	040572	050106	020123	047111	EM112:	.ASCIZ	'FPS INCORRECT AFTER LDFPS @ (RO)+.'
4009	040634	042114	050106	020123	EM113:	.ASCIZ	'LDFPS @ (RO)+ TRAPPED TO 4.'
4010	040667	106	051123	020103	EM114:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS @-(RO).'
4011	040737	106	051520	044440	EM115:	.ASCIZ	'FPS INCORRECT AFTER LDFPS @-(RO).'
4012	041001	114	043104	051520	EM116:	.ASCIZ	'LDFPS @-(RO) TRAPPED TO 4.'

4013	041034	051506	041522	024040	EM117:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS 241(RO).'
4014	041105	106	051520	044440	EM120:	.ASCIZ	'FPS INCORRECT AFTER LDFPS 241(RO).'
4015	041150	042114	050106	020123	EM121:	.ASCIZ	'LDFPS 241(RO) TRAPPED TO 4.'
4016	041204	051506	041522	024040	EM122:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDFPS @241(RO).'
4017	041256	050106	020123	047111	EM123:	.ASCIZ	'FPS INCORRECT AFTER LDFPS @241(RO).'
4018	041322	042114	050106	020123	EM124:	.ASCIZ	'LDFPS @241(RO) TRAPPED TO 4.'
4019	041357	120	020103	047111	EM125:	.ASCIZ	'PC INCORRECT AFTER LDCLD (PC)+,ACO.'
4020	041423	114	041504	042114	EM126:	.ASCIZ	'LDCLD (PC)+,ACO TRAPPED TO 4.'
4021	041461	106	051123	020103	EM127:	.ASCIZ	'FSRC (RO) INCORRECT AFTER LDCLD (RO)+,ACO.'
4022	041534	042522	052523	052114	EM130:	.ASCII	'RESULT OR FPS INCORRECT AFTER LDC'
4023	041575	111	020106	051050	EM130X:	.ASCIZ	'IF (RO),ACO.'
4024	041612	042522	052523	052114	EM131:	.ASCII	'RESULT OR FPS INCORRECT AFTER LDC'
4025	041653	111	020104	051050	EM131X:	.ASCIZ	'ID (RO),ACO.'
4026	041670	042522	052523	052114	EM132:	.ASCII	'RESULT OR STATUS INCORRECT AFTER LDEXP '
4027	041737	050	030122	026051	EM132X:	.ASCIZ	'(RO),ACO.'
4028	041751	106	051504	020124	EM133:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS (RO).'
4029	042017	123	047524	042522	EM134:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS (RO).'
4030	042067	123	043124	051520	EM135:	.ASCIZ	'STFPS (RO) TRAPPED TO 4.'
4031	042120	042106	052123	024040	EM136:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS (RO)+.'
4032	042167	123	047524	042522	EM137:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS (RO)+.'
4033	042240	052123	050106	020123	EM140:	.ASCIZ	'STFPS (RO)+ TRAPPED TO 4.'
4034	042272	042106	052123	024040	EM141:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS -(RO).'
4035	042341	123	047524	042522	EM142:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS -(RO).'
4036	042412	052123	050106	020123	EM143:	.ASCIZ	'STFPS -(RO) TRAPPED TO 4.'
4037	042444	042106	052123	024040	EM144:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS @(RO)+.'
4038	042514	052123	051117	042105	EM145:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS @(RO)+.'
4039	042566	052123	050106	020123	EM146:	.ASCIZ	'STFPS @(RO)+ TRAPPED TO 4.'
4040	042621	106	051504	020124	EM147:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS @-(RO).'
4041	042671	123	047524	042522	EM150:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS @-(RO).'
4042	042743	123	043124	051520	EM151:	.ASCIZ	'STFPS @-(RO) TRAPPED TO 4.'
4043	042776	042106	052123	024040	EM152:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS 241(RO).'
4044	043047	123	047524	042522	EM153:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS 241(RO).'
4045	043122	052123	050106	020123	EM154:	.ASCIZ	'STFPS 241(RO) TRAPPED TO 4.'
4046	043156	042106	052123	024040	EM155:	.ASCIZ	'FDST (RO) INCORRECT AFTER STFPS @241(RO).'
4047	043230	052123	051117	042105	EM156:	.ASCIZ	'STORED DATA INCORRECT AFTER STFPS @241(RO).'
4048	043304	052123	050106	020123	EM157:	.ASCIZ	'STFPS @241(RO) TRAPPED TO 4.'
4049	043341	106	051504	020124	EM160:	.ASCIZ	'FDST (RO) INCORRECT AFTER STCDL ACO,(RO)+.'
4050	043414	042106	052123	024040	EM161:	.ASCIZ	'FDST (RO) INCORRECT AFTER STCDL ACO,-(RO).'
4051	043467	103	047117	042526	EM162:	.ASCII	'CONVERSION OR STATUS INCORRECT AFTER STC'<177>
4052		043537					='. -1
4053		043540					.EVEN
4054	043540	046104	040440	030103	EM162X:	.ASCIZ	'DL ACO,(RO).'
4055	043555	123	047524	042522	EM163:	.ASCII	'STORED EXPONENT OR FPS INCORRECT AFTER STEXP ACO.'
4056	043636	051050	024460	000056	EM163X:	.ASCIZ	'(RO).'
4057	043644	050106	026123	043040	EM164:	.ASCIZ	'FPS, FEC, OR FEA INCORRECT AFTER STST (RO).'
4058	043720	051520	020127	041503	EM165:	.ASCII	'PSW CC BITS <3:0> INCORRECT AFTER STC'<177>
4059		043765					='. -1
4060		043766					.EVEN
4061	043766	046104	040440	047503	EM165X:	.ASCIZ	'DL ACO,(RO).'
4062	044003	120	053523	041440	EM166:	.ASCIZ	'PSW CC BITS <3:0> INCORRECT AFTER STEXP.'
4063							
4064	044054	047125	054105	042520	EM175:	.ASCIZ	'UNEXPECTED FPP TRAP TO 244.'
4065	044110	047125	054105	042520	EM176:	.ASCIZ	'UNEXPECTED CPU TRAP TO 4.'
4066	044142	047125	054105	042520	EM177:	.ASCIZ	'UNEXPECTED CPU TRAP TO 10.'
4067		044176					.EVEN
4068							:

:THESE ARE DATA TABLE HEADERS:					
4069					
4070					
4071	044176	042524	052123	004456	DH1: .ASCII 'TEST.' <tab>'CALL PC.'<tab>'ERROR PC.'</tab></tab>
4072	044226	020011	050106	027123	.ASCIZ <TAB>'EPS.' <tab>'FEC.'</tab>
4073	044243	124	051505	027124	DH2: .ASCIZ 'TEST.' <tab>'CALL PC.'<tab>'ERROR PC.'</tab></tab>
4074					
4075		044243			DH4=DH2
4076		044243			DH5=DH2
4077		044243			DH6=DH2
4078		044243			DH7=DH2
4079		044243			DH10=DH2
4080		044243			DH11=DH2
4081	044274	042524	052123	004456	DH12: .ASCII 'TEST.' <tab>'CALL PC.'<tab>'ERROR PC.'</tab></tab>
4082	044324	041011	042101	050040	.ASCIZ <TAB>'BAD PC' <tab>'EXPD PC'</tab>
4083	044344	042524	052123	004456	DH13: .ASCIZ 'TEST.' <tab>'CALL PC.'<tab>'TRAP PC.'</tab></tab>
4084		044243			DH14=DH2
4085		044243			DH15=DH2
4086		044344			DH16=DH13
4087		044243			DH17=DH2
4088		044243			DH20=DH2
4089		044344			DH21=DH13
4090		044243			DH22=DH2
4091		044243			DH23=DH2
4092		044344			DH24=DH13
4093		044243			DH25=DH2
4094		044243			DH26=DH2
4095		044344			DH27=DH13
4096		044243			DH30=DH2
4097		044243			DH31=DH2
4098		044344			DH32=DH13
4099					
4100		044243			DH34=DH2
4101		044243			DH35=DH2
4102	044374	042524	052123	004456	DH36: .ASCII 'TEST.' <tab>'CALL PC.'<tab>'ERROR PC.'</tab></tab>
4103	044424	041011	042101	043040	.ASCIZ <TAB>'BAD FPS' <tab>'EXPD FPS'</tab>
4104		044243			DH37=DH2
4105		044243			DH40=DH2
4106		044374			DH41=DH36
4107		044243			DH42=DH2
4108		044374			DH43=DH36
4109		044344			DH44=DH13
4110		044243			DH45=DH2
4111		044374			DH46=DH36
4112		044344			DH47=DH13
4113		044243			DH50=DH2
4114		044374			DH51=DH36
4115		044344			DH52=DH13
4116		044243			DH53=DH2
4117		044374			DH54=DH36
4118		044344			DH55=DH13
4119		044243			DH56=DH2
4120		044374			DH57=DH36
4121		044344			DH60=DH13
4122		044243			DH61=DH2
4123		044374			DH62=DH36
4124		044344			DH63=DH13

4125	044243	DH64=DH2
4126	044374	DH65=DH36
4127	044344	DH66=DH13
4128	044374	DH67=DH36
4129	044344	DH70=DH13
4130	044374	DH71=DH36
4131	044344	DH72=DH13
4132	044374	DH73=DH36
4133	044274	DH74=DH12
4134	044344	DH75=DH13
4135	044243	DH76=DH2
4136		
4137	044243	DH100=DH2
4138	044374	DH101=DH36
4139	044344	DH102=DH13
4140	044243	DH103=DH2
4141	044374	DH104=DH36
4142	044344	DH105=DH13
4143	044243	DH106=DH2
4144	044374	DH107=DH36
4145	044344	DH110=DH13
4146	044243	DH111=DH2
4147	044374	DH112=DH36
4148	044344	DH113=DH13
4149	044243	DH114=DH2
4150	044374	DH115=DH36
4151	044344	DH116=DH13
4152	044243	DH117=DH2
4153	044374	DH120=DH36
4154	044344	DH121=DH13
4155	044243	DH122=DH2
4156	044374	DH123=DH36
4157	044344	DH124=DH13
4158	044274	DH125=DH12
4159	044344	DH126=DH13
4160	044243	DH127=DH2
4161	044374	DH130=DH36
4162	044374	DH131=DH36
4163	044243	DH132=DH2
4164	044243	DH133=DH2
4165	044176	DH134=DH1
4166	044344	DH135=DH13
4167	044243	DH136=DH2
4168	044176	DH137=DH1
4169	044344	DH140=DH13
4170	044243	DH141=DH2
4171	044176	DH142=DH1
4172	044344	DH143=DH13
4173	044243	DH144=DH2
4174	044176	DH145=DH1
4175	044344	DH146=DH13
4176	044243	DH147=DH2
4177	044176	DH150=DH1
4178	044344	DH151=DH13
4179	044243	DH152=DH2
4180	044176	DH153=DH1

4181		044344			DH154=DH13	
4182		044243			DH155=DH2	
4183		044176			DH156=DH1	
4184		044344			DH157=DH13	
4185		044243			DH160=DH2	
4186		044243			DH161=DH2	
4187		044243			DH162=DH2	
4188		044374			DH163=DH36	
4189		044374			DH164=DH36	
4190	044446	042524	052123	004456	DH165: .ASCII	'TEST.' <tab>'call pc.'<="" pc.'<tab>'error="" td=""></tab>'call>
4191	044476	020011	050106	027123	.ASCIIZ	<TAB>'FPS.' <tab>'psw.'< td=""></tab>'psw.'<>
4192		044446			DH166=DH165	
4193						
4194		044176			DH175=DH1	
4195		044344			DH176=DH13	
4196		044344			DH177=DH13	
4197		044514			.EVEN	
4198					:	
4199					: THESE ARE THE ERROR MESSAGE DATA TABLES:	
4200					:	
4201	044514	001160	001162	032740	DT1: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4,0
4202	044534	001160	001162	032740	DT2: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ESTAT,\$TMP3,RSTAT,\$TMP4,0
4203						
4204	044556	001160	001162	032740	DT4: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,RZB,\$TMP3,ERZA,\$TMP4,RRZA,\$TMP5,0
4205	044604	001160	001162	032740	DT5: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ACZ,\$TMP3,EXP,\$TMP4,RCD,\$TMP5,0
4206		044604			DT6=DT5	
4207		044556			DT7=DT4	
4208		044604			DT10=DT5	
4209		044604			DT11=DT5	
4210		044514			DT12=DT1	
4211	044632	001160	001162	032740	DT13: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,0
4212		044556			DT14=DT4	
4213		044604			DT15=DT5	
4214		044632			DT16=DT13	
4215		044556			DT17=DT4	
4216		044604			DT20=DT5	
4217		044632			DT21=DT13	
4218		044556			DT22=DT4	
4219		044604			DT23=DT5	
4220		044632			DT24=DT13	
4221		044556			DT25=DT4	
4222		044604			DT26=DT5	
4223		044632			DT27=DT13	
4224		044556			DT30=DT4	
4225		044604			DT31=DT5	
4226		044632			DT32=DT13	
4227						
4228	044644	001160	001162	032740	DT34: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ACZ,\$TMP3,EXP,\$TMP4
4229	044664	033204	001172	033017	.WORD	RCD,\$TMP5,ESTAT,\$TMP6,RSTAT,\$TMP7,0
4230		044534			DT35=DT2	
4231	044702	001160	001162	032740	DT36: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4232	044720	033170	001172	033204	.WORD	EXP,\$TMP5,RCD,\$TMP6,0
4233		044534			DT37=DT2	
4234		044534			DT40 DT2	
4235	044732	001160	001162	032740	DT41: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4236	044750	033234	001172	033170	.WORD	ACOPR,\$TMP5,EXP,\$TMP6,RCD,\$TMP7,0

4237		044556			DT42=DT4		
4238	044766	001160	001162	032740	DT43:	.WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4239	045004	033272	001172	033170		.WORD	DOFR,\$TMP5,EXP,\$TMP6,RCD,\$TMP7,0
4240		044632			DT44=DT13		
4241		044556			DT45=DT4		
4242		044766			DT46=DT43		
4243		044632			DT47=DT13		
4244		044556			DT50=DT4		
4245		044766			DT51=DT43		
4246		044632			DT52=DT13		
4247		044556			DT53=DT4		
4248		044766			DT54=DT43		
4249		044632			DT55=DT13		
4250		044556			DT56=DT4		
4251		044766			DT57=DT43		
4252		044632			DT60=DT13		
4253		044556			DT61=DT4		
4254		044766			DT62=DT43		
4255		044632			DT63=DT13		
4256		044556			DT64=DT4		
4257		044766			DT65=DT43		
4258		044632			DT66=DT13		
4259		044766			DT67=DT43		
4260		044632			DT70=DT13		
4261		044766			DT71=DT43		
4262		044632			DT72=DT13		
4263		044766			DT73=DT43		
4264		044514			DT74=DT1		
4265		044632			DT75=DT13		
4266	045022	001160	001162	032740	DT76:	.WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,DOPR,\$TMP3,EXP,\$TMP4
4267	045042	033204	001172	033017		.WORD	RCD,\$TMP5,ESTAT,\$TMP6,RSTAT,\$TMP7,0
4268							
4269		044556			DT100=DT4		
4270		044514			DT101=DT1		
4271		044632			DT102=DT13		
4272		044556			DT103=DT4		
4273		044514			DT104=DT1		
4274		044632			DT105=DT13		
4275		044556			DT106=DT4		
4276		044514			DT107=DT1		
4277		044632			DT110=DT13		
4278		044556			DT111=DT4		
4279		044514			DT112=DT1		
4280		044632			DT113=DT13		
4281		044556			DT114=DT4		
4282		044514			DT115=DT1		
4283		044632			DT116=DT13		
4284		044556			DT117=DT4		
4285		044514			DT120=DT1		
4286		044632			DT121=DT13		
4287		044556			DT122=DT4		
4288		044514			DT123=DT1		
4289		044632			DT124=DT13		
4290		044514			DT125=DT12		
4291		044632			DT126=DT13		
4292		044556			DT127=DT4		

4293	045060	001160	001162	032740	DT130: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4294	045076	033253	001172	033170	.WORD	SOPR,\$TMP5,EXP,\$TMP6,RCD,\$TMP7,0
4295		045060			DT131=DT130	
4296	045114	001160	001162	032740	DT132: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ACOPR,\$TMP3,SOPR,\$TMP4
4297	045134	033170	001172	033204	.WORD	EXP,\$TMP5,RCD,\$TMP6,ESTAT,\$TMP7,RSTAT,\$TMP10,0
4298		044556			DT133=DT4	
4299	045156	001160	001162	032740	DT134: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3
4300	045172	033170	001170	033204	.WORD	EXP,\$TMP4,RCD,\$TMP5,0
4301		044632			DT135=DT13	
4302		044556			DT136=DT4	
4303		045156			DT137=DT134	
4304		044632			DT140=DT13	
4305		044556			DT141=DT4	
4306		045156			DT142=DT134	
4307		044632			DT143=DT13	
4308		044556			DT144=DT4	
4309		045156			DT145=DT134	
4310		044632			DT146=DT13	
4311		044556			DT147=DT4	
4312		045156			DT150=DT134	
4313		044632			DT151=DT13	
4314		044556			DT152=DT4	
4315		045156			DT153=DT134	
4316		044632			DT154=DT13	
4317		044556			DT155=DT4	
4318		045156			DT156=DT134	
4319		044632			DT157=DT13	
4320		044556			DT160=DT4	
4321		044556			DT161=DT4	
4322	045204	001160	001162	032740	DT162: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,ACOPR,\$TMP3,EXP,\$TMP4
4323	045224	033204	001172	033017	.WORD	RCD,\$TMP5,ESTAT,\$TMP6,RSTAT,\$TMP7,0
4324	045242	001160	001162	032740	DT163: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4325	045260	033234	001172	033311	.WORD	ACOPR,\$TMP5,EXPX,\$TMP6,RCDX,\$TMP7,0
4326	045276	001160	001162	032740	DT164: .WORD	\$TMP0,\$TMP1,HTAB,\$TMP2,HTAB,\$TMP3,\$TMP4
4327	045314	033363	001172	033413	.WORD	ESTAT1,\$TMP5,RSTAT1,\$TMP6,0
4328		044514			DT165=DT1	
4329		044514			DT166=DT1	
4330						
4331		044514			DT175=DT1	
4332		044632			DT176=DT13	
4333		044632			DT177=DT13	
4334					.EVEN	
4335					:	
4336					:THESE ARE FORMAT SPECIFICATIONS FOR THE DATA TABLES:	
4337					:	
4338	045326	004	000	005	DF1: .BYTE	4,0,5,0,5,0,0
4339	045335	004	000	005	DF2: .BYTE	4,0,5,0,5,2,5,2
4340						
4341	045345	004	000	005	DF4: .BYTE	4,0,5,0,5,0,5,0,5,0
4342	045357	004	000	005	DF5: .BYTE	4,0,5,0,5,3,5,2,5,2
4343	045371	004	000	005	DF6: .BYTE	4,0,5,0,5,3,5,2,5,3
4344		045345			DF7=DF4	
4345	045403	004	000	005	DF10: .BYTE	4,0,5,0,5,3,5,3,5,3
4346		045403			DF11=DF10	
4347		045326			DF12=DF1	
4348	045415	004	000	005	DF13: .BYTE	4,0,5,0

4349	045345			DF14=DF4	
4350	045403			DF15=DF10	
4351	045415			DF16=DF13	
4352	045345			DF17=DF4	
4353	045403			DF20=DF10	
4354	045415			DF21=DF13	
4355	045345			DF22=DF4	
4356	045403			DF23=DF10	
4357	045415			DF24=DF13	
4358	045345			DF25=DF4	
4359	045403			DF26=DF10	
4360	045415			DF27=DF13	
4361	045345			DF30=DF4	
4362	045403			DF31=DF10	
4363	045415			DF32=DF13	
4364					
4365	045421	004	000	005	DF34: .BYTE 4,0,5,0,5,3,5,3,5,3,5,2,5,2
4366	045335				DF35=DF2
4367	045437	004	000	005	DF36: .BYTE 4,0,5,0,5,0,0,5,3,5,3
4368	045335				DF37=DF2
4369	045335				DF40=DF2
4370	045452	004	000	005	DF41: .BYTE 4,0,5,0,5,0,0,5,3,5,3,5,3
4371	045345				DF42=DF4
4372	045452				DF43=DF41
4373	045415				DF44=DF13
4374	045345				DF45=DF4
4375	045452				DF46=DF41
4376	045415				DF47=DF13
4377	045345				DF50=DF4
4378	045452				DF51=DF41
4379	045415				DF52=DF13
4380	045345				DF53=DF4
4381	045452				DF54=DF41
4382	045415				DF55=DF13
4383	045345				DF56=DF4
4384	045452				DF57=DF41
4385	045415				DF60=DF13
4386	045345				DF61=DF4
4387	045452				DF62=DF41
4388	045415				DF63=DF13
4389	045345				DF64=DF4
4390	045452				DF65=DF41
4391	045415				DF66=DF13
4392	045452				DF67=DF41
4393	045415				DF70=DF13
4394	045452				DF71=DF41
4395	045415				DF72=DF13
4396	045452				DF73=DF41
4397	045326				DF74=DF1
4398	045415				DF75=DF13
4399	045421				DF76=DF34
4400					
4401	045345				DF100=DF4
4402	045326				DF101=DF1
4403	045415				DF102=DF13
4404	045345				DF103=DF4

4405	045326			DF104=DF1		
4406	045415			DF105=DF13		
4407	045345			DF106=DF4		
4408	045326			DF107=DF1		
4409	045415			DF110=DF13		
4410	045345			DF111=DF4		
4411	045326			DF112=DF1		
4412	045415			DF113=DF13		
4413	045345			DF114=DF4		
4414	045326			DF115=DF1		
4415	045415			DF116=DF13		
4416	045345			DF117=DF4		
4417	045326			DF120=DF1		
4418	045415			DF121=DF13		
4419	045345			DF122=DF4		
4420	045326			DF123=DF1		
4421	045415			DF124=DF13		
4422	045326			DF125=DF12		
4423	045415			DF126=DF13		
4424	045345			DF127=DF4		
4425	045467	004	000	005	DF130: .BYTE 4,0,5,0,5,0,0,5,2,5,2,5,2	
4426	045504	004	000	005	DF131: .BYTE 4,0,5,0,5,0,0,5,2,5,3,5,3	
4427	045521	004	000	005	DF132: .BYTE 4,0,5,0,5,3,5,0,5,3,5,3,5,2,5,2	
4428	045345				DF133=DF4	
4429	045541	004	000	005	DF134: .BYTE 4,0,5,0,5,0,5,3,5,3	
4430	045415				DF135=DF13	
4431	045345				DF136=DF4	
4432	045541				DF137=DF134	
4433	045415				DF140=DF13	
4434	045345				DF141=DF4	
4435	045541				DF142=DF134	
4436	045415				DF143=DF13	
4437	045345				DF144=DF4	
4438	045541				DF145=DF134	
4439	045415				DF146=DF13	
4440	045345				DF147=DF4	
4441	045541				DF150=DF134	
4442	045415				DF151=DF13	
4443	045345				DF152=DF4	
4444	045541				DF153=DF134	
4445	045415				DF154=DF13	
4446	045345				DF155=DF4	
4447	045541				DF156=DF134	
4448	045415				DF157=DF13	
4449	045345				DF160=DF4	
4450	045345				DF161=DF4	
4451	045553	004	000	005	DF162: .BYTE 4,0,5,0,5,3,5,2,5,2,5,2,5,2	
4452	045571	004	000	005	DF163: .BYTE 4,0,5,0,5,0,0,5,3,5,0,5,0	
4453	045606	004	000	005	DF164: .BYTE 4,0,5,0,5,0,0,5,2,5,2	
4454	045326				DF165=DF1	
4455	045326				DF166=DF1	
4456						
4457	045326				DF175=DF1	
4458	045415				DF176=DF13	
4459	045415				DF177=DF13	
4460	045622				.EVEN	

4462
4463
4464
4465
4466
4467 045622 000240
4468 045624 000240
4469 045626 000240
4470 045630 000544
4471 045632 000144
4472 046142 042777 000200 132770 1\$:
4473 046150 104401 046156
(1) 046154 000420
(1)
(1) 046216
4474 046216 000207
4475
4476
4477
4478 046220
4479 000001

```

.SBTTL AREA RESERVED FOR LOGIC ANALYZER SET-UP CODE (NOT INCLUDED).
://////////
: FOR FUTURE EXPANSION, LOGIC ANALYZER SET UP CODE MAY BE PLACED HERE.
: CALLED AT END OF $SCOPE ROUTINE IF SWR<8:7> ARE BOTH SET.
:
ANALYZE: NOP
NOP
NOP
BR 1$
.BLKW 100.
BIC #SW7,@SWR
TYPE .65$
BR 64$
:65$: .ASCIZ 'ANALYZER SET-UP NOT INSTALLED'<CRLF>
64$:
RTS PC
:
://////////
LASTAD-
.END

```

: SETUP CODE CAN GO HERE.
: BUT, SINCE WE HAVE NONE...
:...JUST CLEAR THE 'ANALYZE' BIT...
:...AND SAY SO.
: TYPE ASCIZ STRING
: GET OVER THE ASCIZ

AAABDO	025256	ADDW12=	000000	BBEXP	004454	CCCBDO	025716	DF125 =	045326
AAABSJ	024746	ADDW13=	000000	BBRCD	004444	CCCBSU	025442	DF126 =	045415
AAAB1	024034	ADDW14=	000000	BB1	004264	CCCB1	025264	DF127 =	045345
AAAB10	024364	ADDW15=	000000	BB3	004336	CCCB2	025306	DF13	045415
AAAB11	024414	ADDW2 =	000000	BIT0 =	000001	CCCB3	025330	DF130	045467
AAAB12	024444	ADDW3 =	000000	BIT00 =	000001	CCCB4	025352	DF131	045504
AAAB13	024474	ADDW4 =	000000	BIT01 =	000002	CCCB5	025374	DF132	045521
AAAB14	024524	ADDW5 =	000000	BIT02 =	000004	CCCB6	025416	DF133 =	045345
AAAB15	024554	ADDW6 =	000000	BIT03 =	000010	CHECK2	004052	DF134	045541
AAAB16	024604	ADDW7 =	000000	BIT04 =	000020	CHECK4	004062	DF135 =	045415
AAAB17	024634	ADDW8 =	000000	BIT05 =	000040	CKSWR =	104407	DF136 =	045345
AAAB18	024664	ADDW9 =	000000	BIT06 =	000100	CLRFPS=	104412	DF137 =	045541
AAAB19	024714	ADEVCT=	000000	BIT07 =	000200	CR	- 000015	DF14	= 045345
AAAB2	024064	ADEVVM =	000000	BIT08 =	000400	CRLF	= 000200	DF140 =	045415
AAAB3	024114	AENV =	000000	BIT09 =	001000	DBDONE	005300	DF141 =	045345
AAAB4	024144	AENVVM =	000000	BIT1 =	- 000002	DBEXP	005270	DF142 =	045541
AAAB5	024174	AFATAL=	000000	BIT10 =	002000	DBP1	005250	DF143 =	045415
AAAB6	024224	AMADR1=	000000	BIT11 =	- 004000	DBP2	005260	DF144 =	045345
AAAB7	024254	AMADR2=	000000	BIT12 =	- 010000	DB04	005236	DF145 =	045541
AAAB8	024304	AMADR3=	000000	BIT13 =	- 020000	DB1	005066	DF146 =	045415
AAAB9	024334	AMADR4=	000000	BIT14 =	- 040000	DB2	005162	DF147 =	045345
AABDON	014244	AMAMS1=	000000	BIT15 =	- 100000	DDBDON	015524	DF15	= 045403
AAB04	014160	AMAMS2=	000000	BIT2 =	- 000004	DDB04	015504	DF150 =	045541
AAB1	013764	AMAMS3=	000000	BIT3 =	- 000010	DDB1	015372	DF151 =	045415
AAB10	014172	AMAMS4=	000000	BIT4 =	- 000020	DDB10	015520	DF152 =	045345
AAB11	014202	AMSGAD=	000000	BIT5 =	- 000040	DDDBLJ	026064	DF153 =	045541
AAB12	014212	AMSGLG=	000000	BIT6 =	- 000100	DDDB1	025724	DF154 =	045415
AAB13	014222	AMSGTY=	000000	BIT7 =	- 000200	DDISP	= 177570	DF155 =	045345
AAB14	014232	AMTYP1=	000000	BIT8 =	- 000400	DF1	045326	DF156 =	045541
AAB15	014242	AMTYP2=	000000	BIT9 =	- 001000	DF10	045403	DF157 =	045415
AAB2	014010	AMTYP3=	000000	BPTVEC=	000014	DF100 =	045345	DF16	= 045415
AAB3	014032	AMTYP4=	000000	CBDAT	005050	DF101 =	045326	DF160 =	045345
ABASE =	000000	ANALYZ	045622	CBDONE	005060	DF102 =	045415	DF161 =	045345
ABDONE	004256	APASS =	000000	CBRCD	005030	DF103 =	045345	DF162	045553
ABORT	032775	APRIOR=	000000	CBXPD	005050	DF104 =	045326	DF163	045571
AB1	004130	APTCSU=	000040	CBXPF	005040	DF105 =	045415	DF164	045606
ACDW1 =	000000	APTENV=	000001	CB1	004502	DF106 =	045345	DF165 =	045326
ACDW2 =	000000	APTSIZ=	000200	CB20	004660	DF107 =	045326	DF166 =	045326
ACOPR	033234	APTSPO=	000100	CB23	004740	DF11	= 045403	DF17	= 045345
ACPUOP=	000000	ASWREG=	000000	CB3	004570	DF110 =	045415	DF175 =	045326
ACZ	033220	ATESTN=	000000	CCBDON	015364	DF111 =	045345	DF176 =	045415
ACO =	%000000	AUNIT =	000000	CCB1	014470	DF112 =	045326	DF177 =	045415
AC1 =	%000001	AUSWR =	000000	CCB10	015046	DF113 =	045415	DF2	= 045335
AC2 =	%000002	AVECT1=	000000	CCB2	014522	DF114 =	045345	DF20	= 045403
AC3 =	%000003	AVECT2=	000000	CCB20	015064	DF115 =	045326	DF21	= 045415
AC4 =	%000004	BBBDON	014462	CCB3	014554	DF116 =	045415	DF22	= 045345
AC5 =	%000005	BBB04	014430	CCB30	015102	DF117 =	045345	DF23	= 045403
AC6 =	%000006	BBB1	014252	CCB4	014606	DF12	= 045326	DF24	= 045415
AC7 =	%000007	BBB10	014442	CCB5	014640	DF120 =	045326	DF25	= 045345
ADDW0 =	000000	BBB11	014452	CCB6	014672	DF121 =	045415	DF26	= 045403
ADDW1 =	000000	BBB12	014320	CCB7	014724	DF122 =	045345	DF27	= 045415
ADDW10=	000000	BBDAT	004464	CCB8	014756	DF123 =	045326	DF30	= 045345
ADDW11=	000000	BBDONE	004474	CCB9	015010	DF124 =	045415	DF31	= 045403

DF 32 = 045415	DH112 = 044374	DH177 = 044344	DISPRE 000174	DT153 = 045156
DF 34 = 045421	DH113 = 044344	DH2 = 044243	DOPR 033272	DT154 = 044632
DF 35 = 045335	DH114 = 044243	DH20 = 044243	DSWR = 177570	DT155 = 044556
DF 36 = 045437	DH115 = 044374	DH21 = 044344	DT1 = 044514	DT156 = 045156
DF 37 = 045335	DH116 = 044344	DH22 = 044243	DT10 = 044604	DT157 = 044632
DF 4 = 045345	DH117 = 044243	DH23 = 044243	DT100 = 044556	DT16 = 044632
DF 40 = 045335	DH12 = 044274	DH24 = 044344	DT101 = 044514	DT160 = 044556
DF 41 = 045452	DH120 = 044374	DH25 = 044243	DT102 = 044632	DT161 = 044556
DF 42 = 045345	DH121 = 044344	DH26 = 044243	DT103 = 044556	DT162 = 045204
DF 43 = 045452	DH122 = 044243	DH27 = 044344	DT104 = 044514	DT163 = 045242
DF 44 = 045415	DH123 = 044374	DH30 = 044243	DT105 = 044632	DT164 = 045276
DF 45 = 045345	DH124 = 044344	DH31 = 044243	DT106 = 044556	DT165 = 044514
DF 46 = 045452	DH125 = 044274	DH32 = 044344	DT107 = 044514	DT166 = 044514
DF 47 = 045415	DH126 = 044344	DH34 = 044243	DT11 = 044604	DT17 = 044556
DF 5 = 045357	DH127 = 044243	DH35 = 044243	DT110 = 044632	DT175 = 044514
DF 50 = 045345	DH13 = 044344	DH36 = 044374	DT111 = 044556	DT176 = 044632
DF 51 = 045452	DH130 = 044374	DH37 = 044243	DT112 = 044514	DT177 = 044632
DF 52 = 045415	DH131 = 044374	DH4 = 044243	DT113 = 044632	DT2 = 044534
DF 53 = 045345	DH132 = 044243	DH40 = 044243	DT114 = 044556	DT20 = 044604
DF 54 = 045452	DH133 = 044243	DH41 = 044374	DT115 = 044514	DT21 = 044632
DF 55 = 045415	DH134 = 044176	DH42 = 044243	DT116 = 044632	DT22 = 044556
DF 56 = 045345	DH135 = 044344	DH43 = 044374	DT117 = 044556	DT23 = 044604
DF 57 = 045452	DH136 = 044243	DH44 = 044344	DT12 = 044514	DT24 = 044632
DF 6 = 045371	DH137 = 044176	DH45 = 044243	DT120 = 044514	DT25 = 044556
DF 60 = 045415	DH14 = 044243	DH46 = 044374	DT121 = 044632	DT26 = 044604
DF 61 = 045345	DH140 = 044344	DH47 = 044344	DT122 = 044556	DT27 = 044632
DF 62 = 045452	DH141 = 044243	DH5 = 044243	DT123 = 044514	DT30 = 044556
DF 63 = 045415	DH142 = 044176	DH50 = 044243	DT124 = 044632	DT31 = 044604
DF 64 = 045345	DH143 = 044344	DH51 = 044374	DT125 = 044514	DT32 = 044632
DF 65 = 045452	DH144 = 044243	DH52 = 044344	DT126 = 044632	DT34 = 044644
DF 66 = 045415	DH145 = 044176	DH53 = 044243	DT127 = 044556	DT35 = 044534
DF 67 = 045452	DH146 = 044344	DH54 = 044374	DT13 = 044632	DT36 = 044702
DF 7 = 045345	DH147 = 044243	DH55 = 044344	DT130 = 045060	DT37 = 044534
DF 70 = 045415	DH15 = 044243	DH56 = 044243	DT131 = 045060	DT4 = 044556
DF 71 = 045452	DH150 = 044176	DH57 = 044374	DT132 = 045114	DT40 = 044534
DF 72 = 045415	DH151 = 044344	DH6 = 044243	DT133 = 044556	DT41 = 044732
DF 73 = 045452	DH152 = 044243	DH60 = 044344	DT134 = 045156	DT42 = 044556
DF 74 = 045326	DH153 = 044176	DH61 = 044243	DT135 = 044632	DT43 = 044766
DF 75 = 045415	DH154 = 044344	DH62 = 044374	DT136 = 044556	DT44 = 044632
DF 76 = 045421	DH155 = 044243	DH63 = 044344	DT137 = 045156	DT45 = 044556
DH1 = 044176	DH156 = 044176	DH64 = 044243	DT14 = 044556	DT46 = 044766
DH10 = 044243	DH157 = 044344	DH65 = 044374	DT140 = 044632	DT47 = 044632
DH100 = 044243	DH16 = 044344	DH66 = 044344	DT141 = 044556	DT5 = 044604
DH101 = 044374	DH160 = 044243	DH67 = 044374	DT142 = 045156	DT50 = 044556
DH102 = 044344	DH161 = 044243	DH7 = 044243	DT143 = 044632	DT51 = 044766
DH103 = 044243	DH162 = 044243	DH70 = 044344	DT144 = 044556	DT52 = 044632
DH104 = 044374	DH163 = 044374	DH71 = 044374	DT145 = 045156	DT53 = 044556
DH105 = 044344	DH164 = 044374	DH72 = 044344	DT146 = 044632	DT54 = 044766
DH106 = 044243	DH165 = 044446	DH73 = 044374	DT147 = 044556	DT55 = 044632
DH107 = 044374	DH166 = 044446	DH74 = 044274	DT15 = 044604	DT56 = 044556
DH11 = 044243	DH17 = 044243	DH75 = 044344	DT150 = 045156	DT57 = 044766
DH110 = 044344	DH175 = 044176	DH76 = 044243	DT151 = 044632	DT6 = 044604
DH111 = 044243	DH176 = 044344	DISPLA 001142	DT152 = 044556	DT60 = 044632

DT61 = 044556	EM125 041357	EM24 034711	ERT2 032710	JB2 006472
DT62 = 044766	EM126 041423	EM25 034746	ERT3 032714	JJB DON 016554
DT63 = 044632	EM127 041461	EM26 035021	ERT4 032720	JJB04 016534
DT64 = 044556	EM13 034211	EM27 035067	ERZA 033122	JJB1 016416
DT65 = 044766	EM130 041534	EM30 035125	ESTAT 033017	JJB10 016550
DT66 = 044632	EM130X 041575	EM31 035201	ESTAT1 033363	K - 000200
DT67 = 044766	EM131 041612	EM32 035250	EXP 033170	KBDONE 007530
DT7 = 044556	EM131X 041653	EM34 035307	EXPX 033311	KBSUB 007304
DT70 = 044632	EM132 041670	EM34X 035360	FBA1 005720	KB1 006636
DT71 = 044766	EM132X 041737	EM35 035375	FB0 005706	KB10 007250
DT72 = 044632	EM133 041751	EM36 035466	FBDONE 005742	KB2 006670
DT73 = 044766	EM134 042017	EM36X 035527	FBP1 005732	KB3 006722
DT74 = 044514	EM135 042067	EM37 035537	FB04 005674	KB4 006754
DT75 = 044632	EM136 042120	EM4 033572	FB1 005522	KB5 007006
DT76 045022	EM137 042167	EM40 035623	FB2 005606	KB6 007046
EBA1 005474	EM14 034245	EM41 035707	FFBDON 016044	KB7 007100
EBB0 005464	EM140 042240	EM42 035757	FFB04 016010	KB8 007132
EBDONE 005514	EM141 042272	EM43 036024	FFB1 015674	KB9 007164
EBP1 005504	EM142 042341	EM44 036075	FFB10 016032	KKBDON 016740
EB04 005452	EM143 042412	EM45 036125	FPVEC = 000244	KKB04 016700
EB1 005306	EM144 042444	EM46 036173	GBA1 006146	KKB1 016562
EB2 005364	EM145 042514	EM47 036245	GB0 006134	KKB10 016720
FEBDON 015666	EM146 042566	EM5 033642	GBDONE 006170	KKB11 016730
EEB04 015646	EM147 042621	EM50 036276	GBP1 006160	LASTAD= 046220
EEB1 015532	EM15 034316	EM51 036344	GB04 006122	LASTEM= 000200
EEB10 015662	EM150 042671	EM52 036416	GB1 005750	LASTST= 000062
EMTVEC= 000030	EM151 042743	EM53 036447	GB2 006034	LBDONE 007632
EM1 033443	EM152 042776	EM54 036516	GGBDON 016226	LB1 007536
EM10 034031	EM153 043047	EM55 036571	GGB04 016166	LB2 007554
EM10X 034060	EM154 043122	EM56 036623	GGB1 016052	LF = 000012
EM100 040057	EM155 043156	EM57 036672	GGB10 016206	LLBDON 017062
EM101 040125	EM156 043230	EM6 033705	GGB11 016216	LLB04 017052
EM102 040165	EM157 043304	EM60 036745	GTSWR = 104406	LLB1 016746
EM103 040216	EM16 034362	EM61 036777	HB0 006356	LOOP 003716
EM104 040265	EM160 043341	EM62 037047	HB DONE 006400	LUPERR= 104411
EM105 040326	EM161 043414	EM63 037123	HBP1 006370	MBDB 010036
EM106 040360	EM162 043467	EM64 037156	HB04 006344	MB DONE 010066
EM107 040427	EM162X 043540	EM65 037227	HB1 006176	MBED 010056
EM11 034074	EM163 043555	EM66 037304	HB2 006254	MBEF 010046
EM110 040470	EM163X 043636	EM67 037340	HBDON 016410	MB1 007640
EM111 040522	EM164 043644	EM7 033761	HFB04 016350	MB2 007670
EM112 040572	EM165 043720	EM7X 034015	HFB1 016234	MB3 007716
EM113 040634	EM165X 043766	EM70 037427	HFB10 016370	MBDON 017156
EM114 040667	EM166 044003	EM71 037475	HFB11 016400	MMB1 017070
EM115 040737	EM17 034416	EM72 037564	HT = 000011	NBDONE 010176
EM116 041001	EM175 044054	EM73 037632	HTAB 032740	NB1 010074
EM117 041034	EM176 044110	EM74 037704	IOTVEC= 000020	NBDON 020060
EM12 034140	EM177 044142	EM75 037752	JBA1 006620	NBSUB 017666
EM120 041105	EM2 033516	EM76 040003	JB0 006574	NB1 017164
EM121 041150	EM20 034470	EM76X 040044	JBDONE 006622	NB10 017404
EM122 041204	EM21 034535	ERRVEC= 000004	JBP1 006606	NB11 017424
EM123 041256	EM22 034572	ERTYPE 032400	JB04 006562	NB12 017444
EM124 041322	EM23 034644	ERT1 032524	JB1 006406	NB13 017464

NAB14	017504	QOBDON	022016	SKP2	= 000402	TRP04	003774	TST7	005746
NAB15	017524	QOBSUB	021502	SKP3	- 000403	TRP10	004004	TTBDON	022624
NAB16	017544	QOB1	020600	SOPR	033253	TRP244	003750	TTB1	022430
NAB17	017564	QOB10	021174	SPAC1	032736	TRTVEC=	000014	TYPDS	= 104405
NAB18	017604	QOB11	021230	SSBDON	022422	TST1	004126	TYPE	= 104401
NAB19	017624	QOB12	021264	SSB1	022226	TST10	006174	TYPOC	= 104402
NAB2	017204	QOB13	021320	STACK	- 001100	TST11	006404	TYPON	= 104404
NAB20	017644	QOB14	021354	START	003312	TST12	006626	TYPOS	= 104403
NAB3	017224	QOB15	021410	STKLMT=	177774	TST13	007534	UBDONE	012272
NAB4	017244	QOB16	021444	SVPC	- 030752	TST14	007636	UB04	012172
NAB5	017264	QOB2	020634	SWR	001140	TST15	010072	UB1	011744
NAB6	017304	QOB3	020670	SWREG	000176	TST16	010202	UB10	012204
NAB7	017324	QOB4	020724	SWU	= 000001	TST17	010312	UB11	012214
NAB8	017344	QOB5	020760	SW00	= 000001	TST2	004262	UB12	012224
NAB9	017364	QOB6	021014	SW01	- 000002	TST20	010550	UB13	012234
PBDONE	010306	QOB7	021050	SW02	- 000004	TST21	011066	UB14	012244
PB1	010204	QOB8	021104	SW03	= 000010	TST22	011414	UB15	012262
PIRQ	= 177772	QOB9	021140	SW04	= 000020	TST23	011742	UB2	011770
PIRQVE	= 000240	RBDONE	011062	SW05	= 000040	TST24	012276	UB3	012012
POWERM	032742	RB04	011000	SW06	= 000100	TST25	012634	UUBDON	023030
PPBDON	020572	RB1	010552	SW07	= 000200	TST26	013154	UUB1	022632
PPBSUB	020400	RB10	011012	SW08	= 000400	TST27	013476	VBDONE	012630
PPB1	020066	RB11	011022	SW09	= 001000	TST3	004500	VB04	012526
PPB10	020352	RB12	011032	SW1	- 000002	TST30	013762	VB1	012300
PPB2	020112	RB13	011032	SW10	= 002000	TST31	014250	VB10	012540
PPB3	020136	RB14	011042	SW11	= 004000	TST32	014466	VB11	012550
PPB4	020162	RB2	010576	SW12	= 010000	TST33	015370	VB12	012560
PPB5	020206	RB3	010620	SW13	= 020000	TST34	015530	VB13	012570
PPB6	020232	RCD	033204	SW14	= 040000	TST35	015672	VB14	012600
PPB7	020256	RCDX	033336	SW15	= 100000	TST36	016050	VB15	012616
PPB8	020302	RDCHR	= 104410	SW2	= 000004	TST37	016232	VB2	012324
PPB9	020326	RESTAR	003312	SW3	= 000010	TST4	005064	VB3	012346
PRGSIZ-	000114	RESVEC=	000010	SW4	= 000020	TST40	016414	VVBDON	023236
PRO	= 000000	RRBDON	022220	SW5	= 000040	TST41	016560	VVB1	023036
PR1	- 000040	RRB1	022024	SW6	= 000100	TST42	016744	WBDONE	013150
PR2	- 000100	RRZA	033145	SW7	= 000200	TST43	017066	WB04	013066
PR3	= 000140	RSTAT	033047	SW8	= 000400	TST44	017162	WB1	012636
PR4	= 000200	RSTAT1	033413	SW9	= 001000	TST45	020064	WB10	013100
PR5	= 000240	RZB	033077	TAB	= 000011	TST46	020576	WB11	013110
PR6	= 000300	R6	=%000006	TBDONE	011736	TST47	022022	WB12	013120
PR7	- 000340	R7	=%000007	TBITVE=	000014	TST5	005304	WB13	013130
PS	= 177776	SBDONE	011410	TB04	011644	TST50	022224	WB14	013140
PSW	= 177776	SB04	011316	TB1	011416	TST51	022426	WB2	012662
PWRVEC=	000024	SB1	011070	TB10	011656	TST52	022630	WB3	012704
QBDONE	010544	SB10	011330	TB11	011666	TST53	023034	WWBDON	023442
QB1	010314	SB11	011340	TB12	011676	TST54	023242	WWW1	023244
QB10	010474	SB12	011350	TB13	011706	TST55	023446	YBDONE	013472
QB11	010504	SB13	011360	TB14	011716	TST56	023654	YB04	013406
QB12	010514	SB14	011400	TB2	011442	TST57	024032	YB1	013156
QB13	010524	SB2	011114	TB3	011464	TST6	005520	YB10	013420
QB14	010534	SB3	011136	TKVEC	= 000060	TST60	025262	YB11	013430
QB2	010340	SCOPEX	027602	TPVEC	= 000064	TST61	025722	YB12	013440
QB3	010362	SKP1	- 000401	TRAPVE-	000034	TST62	026072	YB13	013450

YB14	013460	\$BDADR	001122	\$GET42	027070	\$PASS	001242	\$TMP5	001172
YB15	013470	\$BDDAT	001126	\$GTSWR	031270	\$PASTM	001006	\$TMP6	001174
YB2	013202	\$BELL	001224	\$HD =	000003	\$PWAD	032166	\$TMP7	001176
YB3	013224	\$CHARC	030224	\$HIBTS	001000	\$PWRDN	032026	\$TN =	000063
YYBDON	023650	\$CKSWR	031220	\$ICNT	001104	\$PWARMG	032162	\$TPB	001152
YYB1	023450	\$CMTAG	001100	\$ILLUP	032172	\$PWRUP	032100	\$TPFLG	001157
ZBDONE	013756	\$CM3 =	000000	\$INTAG	001135	\$QUES	001230	\$TPS	001150
ZB04	013674	\$CM4 =	000020	\$ITEMB	001114	\$RDCHR	031502	\$TRAP	031656
ZB1	013500	\$CNTLG	031627	\$LFLG	031215	\$RDSZ =	000001	\$TRAP2	031700
ZB10	013706	\$CNTLU	031622	\$LPADR	001106	\$RTNAD	027112	\$TRP =	000013
ZB11	013716	\$CPUOP	001262	\$LPERR	001110	\$SAVR6	032176	\$TRPAD	031712
ZB12	013726	\$CRLF	001231	\$MADR1	001266	\$SCOPE	027120	\$TSTM	001004
ZB13	013736	\$DBLK	030742	\$MADR2	001272	\$SETUP=	000137	\$TSTM	001102
ZB14	013746	\$DEVCT	001244	\$MADR3	001276	\$STUP =	177777	\$TYPDS	030526
ZB2	013524	\$DOAGN	027110	\$MADR4	001302	\$SVLAD	027364	\$TYPE	027674
ZB3	013546	\$DTBL	030732	\$MAIL	001234	\$SVPC =	001000	\$TYPEC	030106
ZFUNC	026542	\$ENDAD	027100	\$MAMS1	001264	\$SWR =	167400	\$TYPEX	030226
ZNGS	026623	\$ENDCT	026744	\$MAMS2	001270	\$SWREG	001256	\$TYPOC	030324
ZOP1	026522	\$ENULL	027114	\$MAMS3	001274	\$SWRMK=	000200	\$TYPON	030340
ZOP2	026532	\$ENV	001254	\$MAMS4	001300	\$SWOBT	027436	\$TYPOS	030300
ZTMR	026520	\$ENVM	001255	\$MBADR	001002	\$TERM =	000026	\$UNIT	001246
ZWI	026551	\$EOP	026710	\$MFLG	031214	\$TESTN	001240	\$UNITM	001010
ZWNI	026574	\$EOPCT	026736	\$MNEW	031645	\$TIMES	001220	\$USWR	001260
ZZBDON	024026	\$ERFLG	001103	\$MSGAD	001250	\$TKB	001146	\$VECT1	001304
ZZB1	023656	\$ERMAX	001115	\$MSGLG	001252	\$TKS	001144	\$VECT2	001306
ZZZ	026230	\$ERROR	032200	\$MSGTY	001234	\$TMP0	001160	\$XOFF =	000023
ZZZDON	026654	\$ERRPC	001116	\$MSWR	031634	\$TMP1	001162	\$XON =	000021
ZZZ1	026322	\$ERRTB	001312	\$SMTYP1	001265	\$TMP10	001200	\$XTSTR	027132
ZZZ1A	026350	\$ERTTL	001112	\$SMTYP2	001271	\$TMP11	001202	\$\$GET4=	000000
ZZZ2	026422	\$ESCAP	001222	\$SMTYP3	001275	\$TMP12	001204	\$\$\$W08=	000063
ZZZ3	026434	\$ETABL	001254	\$SMTYP4	001301	\$TMP13	001206	\$\$TCX	030230
\$APTHD	001000	\$ETEND	001312	\$MXCNT	027434	\$TMP14	001210	\$OFILL	030523
\$ATYC	030776	\$FATAL	001236	\$NULL	001154	\$TMP15	001212	. =	046220
\$ATY1	030752	\$FFLG	031216	\$NWTST=	000001	\$TMP16	001214	.CLRFP	031746
\$ATY3	030760	\$FILLC	001156	\$OCNT	030522	\$TMP17	001216	.LPER	031740
\$ATY4	030770	\$FILLS	001155	\$OMODE	030524	\$TMP2	001164	.SX =	001000
\$AUTOB	001134	\$GDADR	001120	\$OVER	027420	\$TMP3	001166		
\$BASE	001310	\$GDDAT	001124			\$TMP4	001170		

. ABS. 046220 000 CON RO ABS LCL D

ERRORS DETECTED: 0

CJFPBA.BIC,CJFPBA/LI:TOC=SYSMAC/ML,CJFPBA
 RUN-TIME: 36 19 1 SECONDS
 RUN-TIME RATIO: 721/57=12.5
 CORE USED: 26K (51 PAGES)